Aprimorando a visualização e
composição de regras SWRL na Web

Adriano Rívoli da Silva

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Aprimorando a visualização e composição de regras SWRL na Web

Adriano Rívoli da Silva

***Orientador:* Prof. Dr. Dilvan de Abreu Moreira**

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

**USP – São Carlos
Janeiro de 2012**

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

S586a Silva, Adriano Rívolli da
Aprimorando a visualização e composição de regras
SWRL na Web / Adriano Rívolli da Silva; orientador
Dilvan de Abreu Moreira -- São Carlos, 2011.
129 p.

Dissertação (Mestrado - Programa de Pós-Graduação em
Ciências de Computação e Matemática Computacional) --
Instituto de Ciências Matemáticas e de Computação,
Universidade de São Paulo, 2011.

1. CE610.31.7. 2. CE610.28. I. Moreira, Dilvan de
Abreu, orient. II. Título.

Dedico este trabalho
a minha esposa Karina

Agradecimentos

Agradeço primeiramente a Deus, que me deu forças para prosseguir nos momentos em que pensei não ser capaz.

Ao meu orientador, professor Dr. Dilvan de Abreu Moreira, que acreditou no meu potencial, e, com muita competência e paciência me orientou neste trabalho.

Aos meus pais Edinelson e Ilaice, e minha irmã Flávia pelo amor, apoio e confiança depositados em mim, e a todos os meus familiares.

À minha esposa Karina, que sofreu e venceu esse grande desafio ao meu lado.

Aos amigos do ICMC, especialmente ao João Paulo Orlando que me ajudou e dará continuidade a este trabalho.

Aos amigos do IFSP – Campus Salto, que criaram condições para meus estudos. De forma especial, agradeço aos professores Dr. Marcelo Paiva e Dr. Cláudio Yamamoto que deram inúmeras sugestões para este trabalho.

Ao professor Dr. Luciano Pansanato, que me incentivou e ajudou a chegar até o ICMC.

Aos meus amigos Andinei Borba, Flavinho Banderat, Rafael Pereira e Mario Mania.

Enfim, a todas as pessoas que fazem parte da minha vida.

Navegar é preciso, viver não é preciso.
(Fernando Pessoa)

Resumo

A Web Semântica tem como meta fazer com que os conteúdos disponibilizados na Web tenham significado não apenas para pessoas, mas também que possam ser processados por máquinas. Essa meta está sendo realizada com o desenvolvimento e uso de ontologias para criar dados anotados semanticamente. Entre as distintas formas de anotação semântica, a *Semantic Web Rule Language* (SWRL) torna possível criar anotações no formato de regras que combinam regras com conceitos definidos em ontologias, especificadas em *Web Ontology Language* (OWL), para representar conhecimento sobre dados por meio de afirmações condicionais. Todavia, à medida que o número dessas regras crescem, seus desenvolvedores podem enfrentar dificuldades para gerenciá-las adequadamente. Um grande conjunto de regras torna-se difícil de entender e propício a erros, principalmente quando usado e mantido de forma colaborativa. Neste trabalho é apresentado um conjunto de soluções para aprimorar o uso e gerenciamento de regras SWRL, que compreendem o desenvolvimento de novas representações visuais, técnicas de classificação de regras e ferramenta de detecção de erros. Essas soluções resultaram no SWRL Editor, uma ferramenta Web de visualização e composição de regras que roda como um *plug-in* para o Web Protégé. Como estudo de caso, foi utilizada a *Autism Phenologue Rules*, uma ontologia para caracterizar fenótipos de autismo, para exemplificar um conjunto grande e complexo de regras SWRL. A partir desse estudo, uma nova representação visual específica para as regras dessa ontologia foi elaborada, permitindo que um especialista em autismo, sem grandes conhecimentos computacionais, seja capaz de ver e editar regras sem ter de se preocupar com a sintaxe da linguagem SWRL. Os resultados obtidos indicam que o SWRL Editor é uma ferramenta clara e intuitiva, contribuindo para um melhor entendimento, criação e gerenciamento de regras SWRL.

Palavras-chaves: Regras SWRL; Editor de regras; Composição de regras; Visualização de regras; Anotação de dados; Web Semântica.

Abstract

The Semantic Web aims to make web content available not only to people but also to computers using machine-readable formats. This goal is being realized with the development and use of ontologies to create semantically annotated data. Among the different ways to annotate data, the Semantic Web Rule Language (SWRL) enables rule-based annotation that combines rules with ontology concepts, defined using the Web Ontology Language (OWL), to represent knowledge about data as conditional assertions. However, as the number of these rule-based annotations grows, developers face problems when trying to manage them. A large rule set becomes difficult to understand and prone to errors, especially when it is collaboratively maintained. This work presents solutions to improve SWRL rule use and management that include the development of new visual representations, classification techniques and error detection tools. These solutions resulted in the SWRL Editor, a web-based visualization and composition tool for SWRL rules that runs as a Web Protégé plug-in. As a case study, we used the Autism Phenologue Rules, an ontology to characterize autism phenotypes, to exemplify a large and complex SWRL rule set. From this study, a new visual representation, specific for this ontology's rules, has been developed, allowing an expert in autism, without a lot of computational knowledge, to be able to view and edit the rules without having to worry about SWRL syntax. The results obtained indicate that the SWRL Editor is a clear and intuitive tool, contributing for a better understanding and easing the creation and management of SWRL rule sets.

Palavras-chaves: *SWRL rules; Rule editors; Rule composition; Rule visualization; Data annotation; Semantic Web.*

Lista de Figuras

Figura 1 – Tripla RDF (Klyne, Carroll e McBride, 2004)	33
Figura 2 – Tipos de anotações: (A) anotação como asserções; (B) anotação como metadados.	37
Figura 3 – Regra SWRL especificada no formato humano e suas partes.	40
Figura 4 – Editor de regras da ferramenta SWRL Tab.	45
Figura 5 – Ferramenta Axiomé: (A) representação visual; (B) formulário de edição; (C) representação textual; (D) expressão gerada para o agrupamento de uma regra.	47
Figura 6 – Ferramenta ACE View na criação de uma regra SWRL.....	49
Figura 8 – Árvore de decisão utilizando a estrutura de fluxograma (IBM ILOG).....	53
Figura 9 – Diagrama de dependência entre as regras (Axiomé)	54
Figura 10 – <i>Rule flow</i> utilizado para criar e visualizar regras (Visual Rules).....	54
Figura 11 – IDE Eclipse com recursos de <i>highlight</i> (Drools)	55
Figura 12 – <i>Template</i> Web para a criação de uma regra (FICO Blaze Advisor)	56
Figura 13 – Interface interativa (CLIPS)	56
Figura 14 – Visualização de regras em linguagem natural com <i>highlight</i> (FICO Blaze Advisor).....	57
Figura 15 – Principais classes da SWRL Editor API.....	63
Figura 16 – Diagrama de objetos ilustrando como a SWRL Editor API representa os átomos, predicados e argumentos de uma regra	64
Figura 17 – Protótipo da ferramenta SWRL Editor (tela de composição das regras)	65
Figura 18 – Comparativo entre (A) uma regra apresentada no formato original e (B) a representação visual SWRL <i>Highlight</i>	66
Figura 19 – Símbolos utilizados na visualização hierárquica	67
Figura 20 – Comparativo entre (A) a representação visual hierárquica e (B) a representação utilizada pela ferramenta Axiomé.....	68
Figura 21 – Modelo da matriz de característica utilizada na análise de similaridade	69
Figura 22 – <i>Tabs</i> e <i>portlets</i> da ferramenta SWRL Editor	78
Figura 23 – Tela da ferramenta SWRL Editor no modo de visualização	79
Figura 24 – Tela da ferramenta SWRL Editor no modo composição	81
Figura 25 – Distribuição dos átomos nas regras, antecedentes e consequentes	87
Figura 26 – Quantidade de ocorrência dos principais predicados que ocorrem no antecedente das regras da ontologia <i>Autism Phenologue Rules</i>	88

Figura 27 – Quantidade de todos os predicados que ocorrem no consequente das regras da ontologia <i>Autism Phenologue Rules</i>	88
Figura 28 – Representação visual SWRL <i>highlight</i> da regra ConcludeSavantNegative_15756693	91
Figura 29 – Representação visual hierárquica da regra ConcludeSavantNegative_15756693	91
Figura 30 – Representação visual hierárquica da regra ConcludeDelayedPhrases_1185572..	92
Figura 31 – Partes distintas da regra ConcludeDelayedPhrases_1185572	93
Figura 32 – Estrutura comum A para as regras do autismo	93
Figura 33 – Estrutura comum B para as regras do autismo	94
Figura 34 – Representação visual do autismo da regra ConcludeDelayedPhrases_1185572 correspondente a estrutura comum A.....	95

Lista de Tabelas

Tabela 1 – Linguagens para construção de ontologias	31
Tabela 2 – Abordagens de classificação de ontologias.....	31
Tabela 3 – Tipos de átomos SWRL (Hassanpour, O'Connor e Das, 2009).....	39
Tabela 4 – Números e tipos de argumentos suportados pelos tipos de átomos.....	39
Tabela 5 – Cores utilizadas na representação <i>SWRL Highlight</i>	66
Tabela 6 – Matriz de características com os átomos das regras <i>def_has_granddaughter</i> , <i>def_has_grandson</i> e <i>def_has_son2</i>	70
Tabela 7 – Matriz de características com os átomos e predicados das regras <i>def_has_nephew1</i> (original e modificada) e <i>def_has_nephew2</i>	72
Tabela 8 – Valores obtidos na análise de similaridade utilizando átomos, predicados e a combinação de ambos	72
Tabela 9 – Lista de avisos, erros léxicos, sintáticos e semânticos para a linguagem SWRL ..	75
Tabela 10 – Comparativo dos recursos de identificação de erros das ferramentas SWRL	76
Tabela 11 – Comparativo das interfaces e funcionalidade disponíveis nas ferramentas de regras SWRL	82
Tabela 12 – Visão geral das regras disponíveis na ontologia <i>Autism Phenologue Rules</i>	86
Tabela 13 – Distribuição dos tipos de átomos das regras do autismo	89
Tabela 14 – Distribuição dos tipos de argumentos das regras do autismo.....	89
Tabela 15 – Mapeamento dos termos do antecedente com consequente para as estruturas comuns A e B	94
Tabela 16 – Resumo dos cálculos de similaridade entre as regras do conjunto <i>Autism Phenologue Rules</i> , considerando os predicados das regras	98
Tabela 17 - Resumo dos cálculos de similaridade entre as regras do conjunto <i>Autism Phenologue Rules</i> , considerando os átomos das regras	98
Tabela 18 – Agrupamento realizado pela ferramenta Axiomé nas regras do autismo	99
Tabela 19 – Agrupamento por similaridade gerado pela aplicação do algoritmo K-means nas regras do autismo	100

Lista de Siglas e Abreviaturas

ACE	<i>Attempto Controlled English</i>
API	<i>Application Programming Interface</i>
ASD	<i>Autism Spectrum Disorder</i>
BRS	<i>Business Rule System</i>
BMIR	<i>Biomedical Informatics Research</i>
CLIPS	<i>C Language Integrated Production System</i>
CSS	<i>Cascading Style Sheets</i>
DAML	<i>DARPA Agent Markup Language</i>
DL	<i>Description Logic</i>
GWT	<i>Google Web Toolkit</i>
IDE	<i>Integrated Development Environment</i>
IHC	<i>Interação Humano-Computador</i>
JDBC	<i>Java Database Connectivity</i>
JESS	<i>Java Expert System Shell</i>
N3	<i>Notation 3</i>
NDAR	<i>National Database for Autism Research</i>
NIH	<i>National Institutes of Health</i>
OBO	<i>Open Biomedical Ontologies</i>
OIL	<i>Ontology Interchange Language</i>
ORM	<i>Object Role Modeling</i>
OWL	<i>Web Ontology Language</i>
RBS	<i>Rule Based Systems</i>
RDF	<i>Resource Description Framework</i>
RDFS	<i>RDF Schema</i>
RMS	<i>Rule Management System</i>
RuleML	<i>Rule Markup Language</i>
SQL	<i>Structured Query Language</i>
SQWRL	<i>Semantic Query-Enhanced Web Rule Language</i>
SRL	<i>Structured Rule Language</i>
SWRL	<i>Semantic Web Rule Language</i>
TRANSLATOR	<i>TRANSlator from LAnguage TO Rules</i>
UML	<i>Unified Modeling Language</i>
URI	<i>Uniform Resource Identifier</i>
W3C	<i>World Wide Web Consortium</i>
WFA	<i>Workshop de Ferramentas e Aplicações</i>
WTD	<i>Workshop de Teses e Dissertação</i>
XML	<i>eXtensible Markup Language</i>

Sumário

1	INTRODUÇÃO.....	23
1.1	CONTEXTUALIZAÇÃO E MOTIVAÇÃO	24
1.2	OBJETIVOS	26
1.3	ESTRUTURA DO TRABALHO	27
2	REGRAS NA WEB SEMÂNTICA	29
2.1	ONTOLOGIA	29
2.2	WEB SEMÂNTICA	32
2.2.1	RDF	32
2.2.2	OWL.....	34
2.3	ANOTAÇÃO DE DADOS.....	36
2.4	REGRAS SWRL.....	38
2.5	CONSIDERAÇÕES FINAIS	41
3	FERRAMENTAS E INTERFACES DE REGRAS	43
3.1	INTRODUÇÃO	43
3.2	FERRAMENTAS DE REGRAS SWRL	44
3.2.1	SWRL Tab.....	44
3.2.2	Axiomé.....	46
3.2.3	ACE View.....	48
3.3	FERRAMENTAS DE REGRAS DE NEGÓCIO	49
3.4	INTERFACES	51
3.4.1	Tabelas de decisão.....	52
3.4.2	Árvores de decisão.....	52
3.4.3	Diagrama gráfico	53
3.4.4	Editor de texto	55
3.4.5	Formulários e templates.....	55
3.4.6	Interface interativa (shell).....	56
3.4.7	Linguagem natural	57
3.5	OUTROS RECURSOS	58
3.5.1	Nome, descrição e partes de regras.....	58
3.5.2	Representações visuais.....	58
3.5.3	Grupos e categorias.....	59
3.5.4	Componentes de sugestão.....	59
3.5.5	Ícones e símbolos.....	59
3.6	CONSIDERAÇÕES FINAIS	60

4	APRIMORANDO O USO DE REGRAS SWRL	61
4.1	INTRODUÇÃO.....	61
4.2	REPRESENTAÇÃO LÓGICA DAS REGRAS	62
4.3	PROTÓTIPOS DE INTERFACE.....	64
4.4	REPRESENTAÇÕES VISUAIS DE REGRAS	65
4.4.1	<i>SWRL Highlight</i>	65
4.4.2	<i>Visualização Hierárquica</i>	67
4.5	ANÁLISE DE SIMILARIDADE	69
4.6	AGRUPAMENTO.....	73
4.7	DETECÇÃO DE ERROS.....	74
4.8	SWRL EDITOR.....	77
4.8.1	<i>Visualização</i>	78
4.8.2	<i>Composição</i>	80
4.8.3	<i>Comparativo entre Ferramentas</i>	82
4.9	CONSIDERAÇÕES FINAIS	83
5	ESTUDO DE CASO: <i>AUTISM PHENOLOGUE RULES</i>.....	85
5.1	INTRODUÇÃO.....	85
5.2	CONJUNTO DE REGRAS.....	86
5.3	REPRESENTAÇÃO VISUAL DAS REGRAS DO AUTISMO	90
5.3.1	<i>Representação visual do autismo</i>	92
5.4	SIMILARIDADE ENTRE AS REGRAS	97
5.5	AGRUPAMENTO.....	98
5.6	DETECÇÃO DE ERROS.....	101
5.7	CONSIDERAÇÕES FINAIS	104
6	CONCLUSÕES	107
6.1	CONTRIBUIÇÕES.....	107
6.1.1	<i>Visualização de regras SWRL</i>	107
6.1.2	<i>Composição das regras SWRL</i>	109
6.1.3	<i>Colaboração entre os usuários</i>	109
6.1.4	<i>Estudo de caso</i>	110
6.2	PUBLICAÇÕES E PRÊMIOS RELACIONADOS AO TRABALHO	110
6.3	TRABALHOS FUTUROS.....	111
	REFERÊNCIAS.....	113
	APÊNDICE A – REPRESENTAÇÕES VISUAIS ALTERNATIVAS	119
	APÊNDICE B – PROTÓTIPOS DE INTERFACE PARA A FERRAMENTA SWRL EDITOR	123
	APÊNDICE C – ESTRUTURA COMUM IDENTIFICADA NAS REGRAS DO AUTISMO	127

1 Introdução

A Web Semântica, uma extensão da Web atual, tem como principal objetivo fazer com que os conteúdos disponibilizados na Web tenham significado, não apenas para pessoas, mas também que possam ser processados por máquinas (Berners-Lee, Hendler e Lassila, 2001). Para tanto, a Web Semântica faz uso de ontologias para a associação de significado explícito aos dados.

Por sua vez, ontologias surgiram como artefatos para representar, organizar e compartilhar conhecimento em um determinado domínio (Noy e McGuinness, 2001). Diversas áreas do saber têm recorrido ao seu uso, como solução para tais requisitos, impulsionando e promovendo o desenvolvimento de novas técnicas, metodologias, linguagens e ferramentas relacionadas a ontologias. Dentre as linguagens de representação de ontologias, destaca-se, neste trabalho, a *Web Ontology Language* (OWL), a linguagem padrão da Web Semântica (Smith, Welty e McGuinness, 2004). Os termos e conceitos definidos em OWL, além de representar e organizar o conhecimento, podem ser empregados na anotação semântica de dados (Erdmann et al., 2000).

Entre as distintas formas de anotação semântica (Shah e Musen, 2007), a linguagem *Semantic Web Rule Language* (SWRL) é a que torna possível a criação de anotações no formato de regras para representar, organizar e compartilhar o conhecimento de um domínio específico por meio de afirmações condicionais. A SWRL é uma linguagem expressiva de regras que combina cláusulas *Horn* com conceitos definidos em OWL e aumenta a capacidade de inferência sobre os indivíduos (instâncias) em uma base de conhecimento em OWL (Horrocks et al., 2004).

Na perspectiva de um profissional de computação, o formalismo adotado na sintaxe das regras SWRL é simples, contudo, o emprego correto dos conceitos no processo de anotação exige o conhecimento de um especialista no domínio em questão. Por outro lado, para este especialista, o formalismo da linguagem SWRL comumente não lhe é habitual, extrapolando suas habilidades. Como se tornar um especialista em um determinado domínio exige anos de investigação e estudo, restam duas possibilidades neste embate:

1. O trabalho conjunto entre especialista e profissional da computação;
2. O especialista estudar e aprender a linguagem SWRL;

Embora estas possibilidades estejam sujeitas a problemas, no primeiro caso principalmente de comunicação e no segundo a dificuldade de se aprender a lógica e o formalismo da

linguagem, a segunda se mostra menos onerosa e mais factível. Uma estratégia para minimizar os problemas, neste caso, consiste em acrescentar um nível mais alto de abstração, isentando os especialistas de domínio de todos os detalhes da sintaxe das regras em SWRL.

Esse trabalho foi desenvolvido em colaboração com o grupo dos professores Dr. Mark Musen e Dr. Amar Das do *Stanford Center for Biomedical Informatics Research (BMIR)* – *Stanford University School of Medicine*.

1.1 Contextualização e Motivação

A Web Semântica é atualmente uma área de intensa pesquisa dentro de Sistemas Web, com forte apoio do *World Wide Web Consortium (W3C)*, apesar de suas tecnologias ainda não terem sido amplamente adotadas. No entanto, diversas áreas da ciência, que fazem uso de ontologias, estão desenvolvendo soluções com o uso das tecnologias disponíveis na Web Semântica (Shadbolt, Hall e Berners-lee, 2006). Especificamente na área das ciências da vida, o W3C estabeleceu o *Semantic Web Health Care and Life Sciences (HCLS) Interest Group* que tem o objetivo de desenvolver, disseminar e dar suporte ao uso de tecnologias da Web Semântica em saúde, ciências da vida, pesquisa clínica e medicina translacional.

A aplicação de ontologias em experimentos biomédicos gera enorme volume de dados anotados. Com isso, a união da Web Semântica com o grande volume de dados anotados da biomedicina, é benéfica para ambas. O aumento do volume de documentos anotados pode alavancar o uso da Web Semântica e, conseqüentemente, a geração de ferramentas e tecnologias que auxiliem a integração automática de diversos conjuntos de dados, o desenvolvimento de repositórios online e a exploração de características como a interoperabilidade. Por outro lado, o uso de anotações na bioinformática faz com que os dados disponíveis nos repositórios possam ser encontrados com maior precisão pelos usuários, que podem formular pesquisas mais flexíveis (Jonquet, Musen e Shah, 2008), além de possibilitar a representação formal do conhecimento adquirido no desenvolvimento de experimentos (Shah e Musen, 2007).

Contudo, é fundamental que a experiência do usuário especialista da área biomédica seja considerada, de tal modo que o mesmo possa utilizar ferramentas específicas para seu nicho de pesquisa. Essas ferramentas devem empregar terminologias e interfaces adequadas a sua cultura, exigindo o mínimo de conhecimento técnico adicional. Ontologias são difíceis de construir e gerenciar (Noy e McGuinness, 2001; Knublauch et al., 2004) e o processo de anotação de dados exige atenção e dedicação do usuário que deve ter sólido conhecimento do

domínio. A usabilidade das ferramentas de anotação é um fator que tem forte impacto na geração de boas anotações (Bada et al., 2004), o que a torna mais importante neste contexto.

Além disso, os desenvolvedores de regras têm se deparado com alguns problemas, principalmente quando o conjunto de regras se torna grande ou possui regras complexas (Hassanpour, O'Connor e Das, 2009). Para sanar esses problemas, eles precisam de ferramentas para criar, visualizar e gerenciar regras, que possibilitem principalmente:

- **Aquisição do conhecimento** sem inconsistências, ambiguidade e redundância;
- **Visualização de regras e conjunto de regras** de forma a facilitar o entendimento e conhecimento das mesmas;

Tais requisitos motivaram a investigação e desenvolvimento de representações visuais para abstrair os detalhes da sintaxe da linguagem SWRL e técnicas para acrescentar ou melhorar os recursos disponíveis para o gerenciamento de regras. O desenvolvimento deste conjunto de soluções foi conduzido juntamente com um estudo de caso que possibilitou a identificação de novos requisitos, assim como a avaliação das soluções propostas.

Foram utilizadas, na condução do estudo de caso, as regras disponíveis na ontologia *Autism Phenologue Rules*¹ (Tu et al., 2008). O autismo, ou mais especificamente o *Autism Spectrum Disorder* (ASD), é definido como uma série de transtornos complexos no desenvolvimento mental caracterizados por deficiências sociais, dificuldades de comunicação e padrões restritos, repetitivos e estereotipados de comportamento (Frith e Hill, 2003). Essa ontologia de autismo estende algumas ontologias disponíveis no Open *Biomedical Ontologies* (OBO) *Foundry* e utiliza regras SWRL para categorizar fenótipos de autismo (na forma de anotações). Ela representa um grande e complexo conjunto de regras, que está inserido no domínio biomédico, e é mantido por um grupo de pesquisadores especialistas em autismo e informática.

O desenvolvimento da ontologia *Autism Phenologue Rules* (e das regras contidas nela) faz parte do projeto Phenologue que busca colaborativamente adquirir, gerenciar e inferir novas informações a cerca de fenótipos de doenças, a partir de dados experimentais e resultados publicados na literatura (Hassanpour, O'Connor e Das, 2010). O projeto é conduzido pelo professor Amar Das, da *Stanford University* que, juntamente com sua equipe, colaboraram diretamente para o desenvolvimento deste trabalho de mestrado. Com essa parceria, a ferramenta resultante será utilizada pelos pesquisadores do projeto Phenologue no gerenciamento das suas anotações.

¹ Ontologia disponível em: <http://purl.org/autism-ontology/1.0/autism-phenologue-rules.pprj>

1.2 Objetivos

O objetivo central deste trabalho é desenvolver soluções que aprimorem o processo de visualização e composição de regras SWRL. Em especial, permitir que especialistas do domínio biomédico, que utilizam o conjunto de regras *Autism Phenologue Rules*, sejam capazes de visualizar e escrever regras (com o mínimo de conhecimento técnico adicional), sem se deparar com os detalhes técnicos da sintaxe da linguagem SWRL. Para atingir esse objetivo, foram explorados 4 tópicos do trabalho com regras SWRL.

O primeiro tópico explorado é a visualização de regras SWRL, ou seja, como apresentar regras e conjuntos de regras de maneira a facilitar seu entendimento e manutenção. Os objetivos deste trabalho aqui são o desenvolvimento de representações visuais que abstraíam a sintaxe da linguagem SWRL; de um método de agrupamento que considere a similaridade das regras; e a geração de informações automáticas, que forneçam uma visão geral e específica, de regras e de conjunto de regras.

O segundo tópico explorado é a composição das regras SWRL, para permitir que os desenvolvedores tenham um controle maior sobre o que é produzido. Nesse caso, os nossos objetivos são a elaboração de uma lista com os possíveis erros (classificados em escopo e tipo) que os desenvolvedores estão sujeitos e implementação de um mecanismo que faça a detecção desses erros. Além disso, foram criadas representações visuais para auxiliar o processo de criação e edição de regras.

O terceiro tópico explorado é a colaboração entre os usuários. O objetivo aqui é permitir o desenvolvimento colaborativo e paralelo das regras SWRL. Para isso, foi desenvolvida uma ferramenta Web, como um *plug-in* do Web Protégé, que integra as soluções desenvolvidas e forneça aos desenvolvedores de regras um ambiente completo de criação e gerenciamento de regras SWRL.

O quarto, e final, tópico é um estudo de caso. O objetivo foi conduzir um estudo aplicando as técnicas desenvolvidas às regras da ontologia *Autism Phenologue Rules* para identificar os padrões, problemas e características desse conjunto de regras. Com isso, podemos validar algumas das técnicas empregadas e contribuir com os pesquisadores biomédicos envolvidos em seu desenvolvimento.

1.3 Estrutura do Trabalho

Este trabalho está organizado da seguinte forma: no capítulo seguinte (Capítulo 2, Regras na Web Semântica) é apresentada a fundamentação teórica para o desenvolvimento deste trabalho, que inclui conceitos de ontologias, Web Semântica, anotação semântica de dados e regras SWRL; no Capítulo 3 (Ferramentas e Interfaces de Regras) são explorados os trabalhos relacionados, que compreendem as ferramentas de regras, suas interfaces e principais recursos utilizados; no Capítulo 4 (Aprimorando o uso de regras SWRL) são descritos os recursos desenvolvidos nesta pesquisa, que consistem em representações visuais e um conjunto de técnicas para auxiliar a visualização e composição de regras, juntamente com a ferramenta elaborada que integra tais recursos; no Capítulo 5 (Estudo de Caso: *Autism Phenologue Rules*) é descrita a aplicação dessas técnicas ao conjunto de regras do autismo, o que possibilitou a realização de uma validação de alguns dos recursos desenvolvidos e uma análise as regras da ontologia; por fim, no Capítulo 6 (Conclusões) são apresentadas as considerações finais deste trabalho, juntamente com as contribuições alcançadas e direcionamentos para trabalhos futuros.

2 Regras na Web Semântica

Neste capítulo, são apresentados os conceitos que fundamentaram esta pesquisa. Para tanto, foi conduzida uma revisão bibliográfica sobre ontologias, Web Semântica, anotação semântica de dados e regras SWRL.

O estudo desenvolvido serviu para estabelecer as convenções adotadas nesta dissertação, além disso, a compreensão destes conceitos foi essencial para o desenvolvimento deste trabalho de mestrado.

2.1 Ontologia

O termo ontologia é emprestado da filosofia (Gruber, 1993) e empregado na ciência da informação e em inteligência artificial, de modo geral, como uma forma de representação do conhecimento (Smith e Welty, 2001). São diversas as definições, tipos, propostas de aplicações e metodologias de construção encontradas na literatura para o termo (Almeida e Bax, 2003).

Gruber (1993) define ontologia como uma especificação explícita da conceituação de um domínio, visando uma classificação padronizada e formal por meio de um vocabulário controlado que representa semanticamente e exatamente os termos de um domínio específico e seus relacionamentos.

Uschold e Grüninger (1996) relatam que o termo ontologia é usado para referenciar o entendimento compartilhado de algum domínio de interesse, que pode ser usado como um *framework* para resolver problemas de interoperabilidade, de reuso e de especificação de dados. Em um trabalho posterior, Uschold (1998) acrescenta que o termo ontologia pode ser empregado como um meio para estruturar uma base de conhecimento, como parte de uma base de conhecimento ou como um mecanismo de interlíngua, integrando repositórios de dados.

Discussões e estudos aprofundados sobre a definição e conceituação do termo ontologia podem ser encontrados em Guarino e Ginarreta (1995); Chandrasekaran, Josephson e Benjamins (1999); Smith e Welty (2001); Almeida e Bax (2003) e Corazzon (2010). Mesmo sem um consenso sobre sua definição, ontologias compartilham características comuns e

impulsionam o desenvolvimento de diversos trabalhos referentes a metodologias, ferramentas, linguagens e aplicações.

Uma ontologia é especificada por meio de componentes básicos que são as **classes**, **relações**, **axiomas** e **instâncias**, além de ser expressa por meio de uma linguagem de construção (Almeida e Bax, 2003).

As **classes**, o foco da maioria das ontologias, são utilizadas para descrever os conceitos de um domínio, possibilitando a organização das classes em um sistema lógico e hierárquico contendo subclasses que representam conceitos mais específicos (Noy e McGuinness, 2001).

As **relações** representam o tipo de interação entre os conceitos de um domínio e as propriedades presentes nas classes e indivíduos. Os termos *slot*, *role*, propriedade e até mesmo atributo podem ser empregados como sinônimo para as relações (Noy e McGuinness, 2001; Horridge et al., 2004). As relações podem ter características próprias como serem transitivas, simétricas, ou terem uma cardinalidade definida.

Os **axiomas** são utilizados para modelar regras assumidas como verdadeiras no domínio em questão, de modo que seja possível associar o relacionamento entre os indivíduos, além de fornecer características descritivas e lógicas para os conceitos. Para Uschold e Grüninger (1996) os axiomas são especificados para definir a semântica e significado dos termos (classes e propriedades) e sugere que a fase de definição dos axiomas (especificação da ontologia) é a mais difícil na construção de ontologias.

Por fim, os **indivíduos**, ou instâncias das classes, são utilizados para representar elementos específicos, ou seja, os próprios dados, que juntamente com a definição de uma ontologia constituem a base de conhecimento (Noy e McGuinness, 2001). Os indivíduos representam objetos do domínio de interesse (Horridge et al., 2004).

Os componentes básicos de uma ontologia são definidos por meio de uma linguagem de representação. Horridge et al. (2004) apontam que diferentes linguagens para ontologias proporcionam diferentes facilidades e recursos. Alguns exemplos de linguagens que se prestam à construção de ontologias são apresentados a seguir, na Tabela 1. Uma listagem mais completa de linguagens de representação de ontologias pode ser encontrada em Su e Ilebrikke (2002) e Almeida e Bax, (2003).

No que se refere à classificação de ontologias, encontram-se abordagens relacionadas com a função, a aplicação, o grau de formalismo do vocabulário, a estrutura e ao conteúdo da conceitualização (Almeida e Bax, 2003). São sintetizadas na Tabela 2 as abordagens referentes ao grau de formalismo e estrutura da conceitualização, adotados como referência neste trabalho.

Tabela 1 – Linguagens para construção de ontologias

Linguagens	Breve descrição
DAML	<i>DARPA Agent Markup Language</i> (DAML) é precursora do DAM+OIL que originou a OWL. Sua sintaxe permite que o computador possa fazer as mesmas inferências simples que os seres humanos podem fazer (Ouellet e Ogbuji, 2002).
OBO	<i>Open Biological Ontologies</i> (OBO) é uma linguagem de ontologia que tem sido frequentemente utilizada para a modelagem de ontologias nas áreas biomédicas. Faz uso de uma sintaxe textual simples que foi projetada para ser compacta, legível por humanos e fácil de ser processada (Golbreich et al., 2007).
OIL	<i>Ontology Interchange Language</i> (OIL) é base para as linguagens da Web Semântica é precursora do DAM+OIL que originou o OWL. É uma linguagem baseada em <i>frames</i> juntamente com as características de lógica descritiva (<i>Description Logic</i> – DL) (Fensel et al., 2001).
OWL	<i>Web Ontology Language</i> (OWL) projetada para atender a necessidade de uma linguagem de representação de ontologias para a Web. Utiliza RDF para a sintaxe e lógica descritiva para a definição de regras de inferência. É o padrão adotado no W3C como linguagem da Web Semântica (McGuinness e Harmelen, 2004).
RDF	<i>Resource Description Framework</i> (RDF) é uma linguagem para definição de informação na Web. Descreve os dados (significado dos termos e conceitos) em um formato que máquinas podem processar. Utiliza o <i>eXtensible Markup Language</i> (XML) e <i>Uniform Resource Identifier</i> (URI) e conjuntos de triplas (sujeito, verbo e objeto) formam as sentenças elementares (Berners-lee, Hendler e Lassila, 2001).

Tabela 2 – Abordagens de classificação de ontologias

Abordagem	Classificação	Descrição
Grau de formalismo (Uchold e Grüninger, 1996)	Altamente informais	Expressadas livremente em linguagem natural.
	Semi-informais	Expressadas em linguagem natural, entretanto com um vocabulário restrito e estruturado diminuindo a ambiguidade e mantendo a clareza.
	Semiformais	Expressadas formalmente em uma linguagem artificial.
	Rigorosamente formal	Termos definidos minuciosamente com semântica formal, teoremas e provas, se tornando uma linguagem sólida.
Estrutura (Haav e Lubi, 2001)	Alto nível	Independente de um problema particular ou domínio, utilizada para descrever conceitos gerais.
	Domínio	Vocabulário referente a um domínio particular ou genérico, especializando os termos definidos em alto nível.
	Tarefa	Vocabulário referente a tarefas ou atividades genéricas, especializando os termos definidos em alto nível.

Por fim, as ontologias tem base em diferentes modelos lógicos possibilitando que os conceitos possam ser bem definidos e descritos, além disso, os modelos lógicos permitem usar um *reasoner* que pode checar a consistência das definições e realizar novas inferências (Horridge et al., 2004).

2.2 Web Semântica

A Web Semântica, uma extensão da Web atual, é uma representação capaz de associar significados explícitos aos conteúdos dos documentos disponíveis na Internet, sendo que sua principal meta é possibilitar que programas processem e interpretem automaticamente esses documentos (Berners-Lee et al., 2001). Para Berners-Lee, a Web Semântica deve possibilitar que computadores sejam capazes de acessar dados estruturados e de definir regras de inferências, transformando grandes volumes de dados em informação.

Seu emprego é fortemente recomendado pelo W3C, que busca desenvolver padrões, arquiteturas de metadados e linguagens para ontologias que juntos possibilitem a integração e entendimento dos dados por computadores, agregando aos mesmos significados. Sua exploração é motivada pelo potencial que tem em transformar a Internet, vista hoje como um repositório de dados, em um repositório explícito de conhecimento, disponível tanto para pessoas como para máquinas. O papel do W3C no contexto da Web atual é o desenvolvimento de padrões, recomendações e orientações com o objetivo de levar a Web ao seu potencial máximo. Além dos avanços relacionados com as aplicações da Web, o W3C tem mobilizado grandes esforços e iniciativas para o desenvolvimento de uma Web para todos, em todos os dispositivos, baseada no conhecimento, com confiança e confiabilidade (Diniz e Cecconi, 2008).

A adoção das tecnologias da Web Semântica e de ontologias na representação de dados, embora não tenha sido amplamente difundida e adotada até o momento (Shadbolt, Hall e Berners-lee, 2006), conta com o apoio de inúmeras empresas na divulgação e desenvolvimento de soluções que fazem o uso da Web Semântica (Feigenbaum et al., 2007).

A tarefa de associar significados aos dados é possível pelo uso de tecnologias como RDF e OWL. O RDF utiliza XML e URI para proporcionar uma representação minimalista do conhecimento na Web e tem como característica principal ser simples. Por outro lado, a OWL é uma tecnologia complexa e voltada para a representação de objetos que requerem grande poder de expressividade. OWL usa RDF e possibilita a criação de ontologias para representação de conhecimento (Shadbolt, Hall e Berners-lee, 2006).

2.2.1 RDF

Resource Description Framework (RDF) é um esquema para a definição de informações na Web, provendo tecnologia para expressar o significado de termos e conceitos de modo que

os computadores possam facilmente processá-los (Berners-Lee et al., 2001). Sua sintaxe pode ser definida com o uso de marcações XML e URI, empregadas para especificar entidades, conceitos, propriedades e relacionamentos. Lassila e Swick (2004) definem que RDF é a tecnologia base para o processamento de metadados, provendo assim interoperabilidade entre aplicações que trocam e processam informações na Web. Para os autores, um dos objetivos do RDF é possibilitar a criação de semântica para dados baseados em XML. Contudo, o principal objetivo do desenvolvimento do RDF é permitir a descrição de recursos de qualquer domínio específico.

Definida pelo W3C, o RDF organiza o conhecimento por meio da ideia de redes semânticas, e permite a representação de conceitos, taxonomia de conceitos e relações binárias (Almeida e Bax, 2003). Em um relato histórico, Smith, Welty e McGuinness (2004) afirmam que o RDF foi a primeira linguagem especificada pelo W3C para representar informações semânticas na Web.

O modelo de dados em RDF é uma forma neutra de representar as expressões utilizadas para atribuir significado aos dados. O modelo básico consiste em três tipos de objetos: recursos, propriedades e afirmações² que são chamadas respectivamente de sujeito, verbo e objeto, formando a tripla RDF (Berners-Lee et al., 2001). A codificação do conhecimento se concretiza em conjuntos de triplas, representadas graficamente na Figura 1.

Figura 1 – Tripla RDF (Klyne, Carroll e McBride, 2004)

O sujeito compreende qualquer coisa que pode ser expressa e descrita em RDF. Alguns exemplos de recursos são: um documento HTML; uma parte de uma página Web; um elemento XML específico; um conjunto de páginas; e objetos que não podem ser acessados diretamente na Web, como um livro impresso ou uma pessoa. Os recursos são sempre nomeados por URI, pois a expressividade deste recurso possibilita que qualquer entidade possa ter seu identificador (Lassila e Swick, 1999).

O verbo consiste nas características, atributos ou relacionamentos utilizados para descrever um recurso. Cada propriedade tem um significado específico, define seus valores permitidos, os tipos de recursos que podem descrever, e sua relação com outras propriedades. O valor de uma propriedade atribuída a um recurso específico é o objeto. O valor da propriedade pode

² Livre tradução para o termo *statement*

ser outro recurso, especificado por um URI, ou um valor literal (simples string ou outro tipo primitivo definido em XML). Tendo como base a tripla, a linguagem RDF possui uma sintaxe abstrata que pode ser expressa em XML, *Notation 3* (N3) ou graficamente, de modo que a semântica formal é definida com base nesta sintaxe abstrata. Uma completa descrição desta sintaxe é apresentada em Klyne, Carroll e McBride (2004).

Uma poderosa extensão do RDF é o RDF *Schema*, usado para descrição de vocabulário utilizado para descrever os relacionamentos internos de propriedades e valores, sendo considerado um mecanismo de extensão semântica para o RDF (Brickley, Guha e McBride, 2004). O RDF *Schema* prove mecanismos para descrever grupos de recursos relacionados, além do relacionamento entre estes recursos. É possível utilizar o conceito de classes, indivíduos, propriedades e subpropriedades semelhantemente ao recurso de hierarquia de classes.

Para Feigenbaum et al. (2007), o RDF é o mais fundamental bloco de construção para a Web Semântica, pois, além de poder ser utilizado para criar dados semânticos, é também utilizado como base para as linguagens de ontologia da Web Semântica. Entretanto para Staab, Maedche e Handschuh (2001), os dados em RDF são fracamente interligados, de modo que a Web Semântica necessita de técnicas mais sofisticadas.

2.2.2 OWL

A linguagem de ontologia *Web Ontology Language* (OWL) é destinada para os que desejam grande expressividade na descrição dos objetos e seus relacionamentos (Shadbolt, Hall e Berners-lee, 2006). O RDF *Schema* é uma linguagem ontológica considerada leve (Staab, Maedche e Handschuh, 2001), pois não contém a complexidade e riqueza de outras linguagens como OWL. Todavia, OWL usa as ligações RDF e trabalha uma camada acima do mesmo (Feigenbaum et al., 2007).

O W3C publicou em 2004 a linguagem de marcação semântica OWL como recomendação para representar e compartilhar ontologias na Web (Smith, Welty e McGuinness, 2004). Contudo, OWL foi concebida de uma revisão da linguagem DAML+OIL incorporando as lições aprendidas durante o desenvolvimento e aplicação desta (McGuinness e Harmelen, 2004).

A OWL 1 provê três sublinguagens com expressividade crescente: OWL *Lite*, OWL DL e OWL *Full*, sendo a primeira a mais simples e a última a mais complexa. A simplicidade e

complexidade referidas aqui dizem respeito ao vocabulário, recursos e capacidades de expressividade de cada sublinguagem.

OWL *Lite* suporta necessidades primárias, como classificação de hierarquia e construções simples, entretanto possui restrições de cardinalidade e tem um número menor de propriedades. O uso desta sublinguagem é recomendado para migração de *thesauri* e outras taxonomias. OWL DL e OWL *Full* compartilham a máxima expressividade incluindo todas as construções da linguagem OWL. A diferença entre estas sublinguagens consiste que OWL DL garante que a ontologia seja sempre computável e decidível enquanto que, com OWL *Full*, não se pode ter esta garantia. Isso ocorre pelo fato de OWL DL respeitar as restrições de definição presentes na lógica *Description Logic*, daí o sufixo DL.

Baader e Nutt (2003) definem *Description Logic* como um formalismo matemático que representa o conhecimento de um domínio de aplicação (o “mundo”) primeiramente definindo os conceitos relevantes desse domínio (sua terminologia) e depois usando esses conceitos para especificar propriedades de objetos e indivíduos que ocorrem nesse domínio (a descrição do mundo). Esse formalismo é que garante que ontologias em OWL DL vão ser computáveis e decidíveis. A finalidade de uma linguagem de representação de conhecimento que usa *Description Logic* vai além de armazenar e definir conceitos e afirmações. Suas definições podem ser validadas (checagem de consistência) e podem conter conhecimento implícito, obtido através de inferências lógicas. Por essa razão, OWL-DL é o foco desse trabalho e as menções a linguagem OWL nesse documento serão para a sublinguagem OWL DL, exceto quando explicitamente expresso o contrário.

A ferramenta capaz de gerar as inferências lógicas e checar a consistência em OWL é chamada de *reasoner*³. Um dos principais testes que essa ferramenta pode realizar é verificar se uma classe é uma subclasse de outra classe, de modo que isso ajuda a manter uma hierarquia correta e identificar classes inconsistentes (Horridge et al., 2004). Em OWL, uma inconsistência acontece se uma classe não pode ter instâncias.

Para Smith, Welty e McGuinness (2004), uma das vantagens do uso da linguagem OWL para a construção de ontologias é que existem ferramentas de classificação. Estes tipos de ferramenta proporcionam um suporte genérico, aplicável a todos os domínios de aplicação. Uma lista de classificadores é encontrada em Grau et al. (2008).

OWL trabalha com suposições de mundo aberto, considerando que as descrições de recursos podem estar presentes em mais de um único arquivo ou escopo. Isso implica na

³ Horridge et al. (2004) sugere que *reasoners* são também conhecidos como classificadores.

impossibilidade de assumir que algo não existe até que isso seja explicitamente definido, ou ainda, algo não pode ser definido como verdadeiro, pelo fato de não poder ser assumido como falso, pois o conhecimento pode existir sem ter sido adicionado à base de conhecimento (Horridge et al., 2004).

Proporcionando uma visão geral da OWL, McGuinness e Harmelen (2004) apresentam a sintaxe expressa em RDF/XML. Com isso, alguns termos possuem o prefixo *rdf:* ou *rdfs:* compreendendo os mesmos termos presentes em RDF ou RDF *Schema* respectivamente.

Recentemente o W3C agregou novos recursos à linguagem OWL dando origem a uma nova recomendação, chamada OWL 2, que é uma extensão e revisão da anterior (W3C OWL Working Group, 2009).

A nova linguagem é consequência de um amadurecimento e evolução da Web Semântica, e com isso, novos recursos e possibilidades foram incorporados em OWL 2. O uso de outras linguagens além do RDF/XML para definição da sintaxe, novos recursos semânticos como redes de propriedades, definição de intervalo de dados, tipos de dados mais ricos e novos tipos de propriedades são exemplos dos novos recursos incorporados.

A justificativa encontrada por Grau et al. (2008) para justificar a necessidade da evolução da OWL, é o seu sucesso. Pois, devido ao sucesso da “OWL 1” foram identificados problemas quanto ao *design* da linguagem, sendo alguns de natureza crítica. Problemas como as limitações de expressividade juntamente com o fato da linguagem OWL ter sua sintaxe influenciada pelo paradigma de *frames*, são alguns dos problemas identificados pelo autor que justificam a evolução da linguagem.

2.3 Anotação de Dados

A tarefa de anotar dados semanticamente consiste em associar significados explícitos a documentos ou conjuntos de dados por meio da utilização de *tags* semânticas, provenientes de ontologias. Kiryakov et al. (2004) definem anotações como a geração específica de metadados com o objetivo de criar novos métodos de acesso à informação ou para ampliar os existentes. Contudo, Shah e Musen (2007) destacam dois diferentes significados para o emprego do termo anotação na biomedicina: anotações como asserções e anotações como metadados.

Anotações como asserções (afirmações ou declarações) são cruciais para ligar os resultados de experimentos com a teoria, aumentando significativamente a qualidade da informação. Este tipo de anotação torna o conhecimento explícito, contudo requer o

envolvimento de um especialista⁴, que, após analisar e interpretar um documento (artigo ou resultado de um experimento) gera informações relevantes e anota o conhecimento extraído do documento. O formato deste tipo de anotação é uma expressão lógica que utiliza os termos de uma ou mais ontologias e se torna um axioma em uma base de conhecimento.

Os desafios atuais para este tipo de anotação, no âmbito da biomedicina, compreendem o desenvolvimento de ferramentas integradas com ontologias que ofereçam recursos de controle das anotações realizadas; a criação de um formalismo sistemático para a geração de afirmações ricas; e, o desenvolvimento de uma metodologia que mensure o esforço humano necessário para a realização destas anotações em bases de dados.

Anotações como metadados são mais frequentes e consistem no uso de declarações, palavras chaves, rótulos ou *tags* que descrevem um documento ou conjunto de dados. Este tipo de anotação pode ser realizado tanto por processos manuais como semiautomáticos ou automáticos, de modo que estes últimos têm como principal desafio, gerar marcações consistentes. O uso deste tipo de anotação possibilita que buscadores e repositórios de dados sejam mais eficientes, pois permitem a realização de buscas semânticas do termo especificado com base no conhecimento expresso em uma ontologia, tornando os resultados mais precisos.

Na Figura 2 são ilustrados os tipos de anotações: anotações como asserções (A) e anotações como metadados (B).

Figura 2 – Tipos de anotações: (A) anotação como asserções; (B) anotação como metadados.

Uma das principais dificuldades quanto à realização de ambos os tipos de anotações é o correto emprego dos termos ontológicos que devem ser aplicados na definição das mesmas. As ontologias de domínio são geralmente extensas, de modo que, durante a anotação o usuário precisa localizar um termo específico entre centenas e até milhares de termos organizados hierarquicamente.

⁴ Na área biomédica, é frequentemente encontrado o termo *curator* (Shah e Musen, 2007)

Note que o uso do termo anotação é também utilizado para descrever um recurso disponível na linguagem OWL, que consiste na possibilidade de atribuir descrições textuais aos elementos da ontologia, como observado em McGuinness e Harmelen (2004) e Grau et al. (2008). Entretanto, o emprego do termo com este significado difere completamente do significado que “anotação” tem neste trabalho.

Atualmente, não existe uma única forma padronizada para descrever uma anotação, tornando o armazenamento e processamento muito mais árduo (Eilbeck e Lewis, 2004). Erdmann et al. (2000) enfatizam que XML e RDF são tecnologias fundamentais para a criação de anotações, além da utilização das linguagens de ontologia e texto. Neste trabalho, o termo anotação é empregado no sentido de anotações como asserções, sendo a linguagem de regras SWRL o formalismo adotado para representação e descrição das anotações.

2.4 Regras SWRL

SWRL é uma expressiva linguagem de regras que combina cláusulas *Horn* com conceitos definidos em OWL e pode ser usada para aumentar a capacidade de inferência sobre os indivíduos em uma base de conhecimento em OWL (Horrocks et al., 2004). Regras em SWRL são compostas de duas partes: o antecedente (*body*) e o consequente (*head*). Cada regra é uma implicação entre o antecedente e o consequente, que pode ser entendida como: quando as condições do antecedente são verdadeiras, então as condições do consequente também são verdadeiras. Ambas as partes consistem em uma conjunção de zero ou mais átomos, não permitindo disjunções ou negação.

Os átomos, por sua vez, são formados por um predicado e um ou mais argumentos (o número e o tipo destes argumentos são determinados pelo tipo do átomo, que por sua vez, é definido pelo predicado utilizado). Embora a especificação W3C defina sete tipos de átomos, neste trabalho foi adotada a convenção usada em Hassanpour, O'Connor e Das (2009), que trabalha com seis tipos de átomos, pois, *SameAs* e *DifferentFrom* embora possuam semânticas diferentes, sintaticamente são idênticos. Na Tabela 3 são apresentados os tipos de átomos e suas descrições.

Os tipos *Class*, *Object property* e *Datavalued property* correspondem a elementos (conceitos) definidos na ontologia. *Data range* correspondem aos tipos de dados utilizados na ontologia. O tipo *Same/different* é utilizado para explicitar a igualdade ou diferença entre dois indivíduos, exigida pelo fato de OWL trabalhar com o paradigma de mundo aberto. Por fim, os *Built-ins* são tipos que encapsulam as mais diversas funções, podendo inclusive ser

estendidos pelos usuários. Cada tipo de átomo define o número e o tipo de argumentos suportados, conforme ilustrado na Tabela 4.

Tabela 3 – Tipos de átomos SWRL (Hassanpour, O'Connor e Das, 2009)

Tipo de átomo	Descrição
<i>Class</i>	O predicado corresponde a uma classe definida na ontologia e recebe um indivíduo ⁵ como argumento.
<i>Object property</i>	O predicado corresponde a uma propriedade definida na ontologia e recebe dois indivíduos como argumentos e os relaciona entre si.
<i>Data valued property</i>	O predicado corresponde a uma propriedade definida na ontologia e recebe dois argumentos: um indivíduo e um valor ⁶ relacionando-os.
<i>Data range</i>	O predicado corresponde a um tipo de dado definido na ontologia e recebe um valor como argumento.
<i>Same/different</i>	O predicado corresponde ao termo “ <i>same as</i> ” ou “ <i>different from</i> ” e recebe dois indivíduos como argumentos definindo que estes são iguais ou diferentes respectivamente.
<i>Built-in</i>	O predicado corresponde a um conjunto de funções pré-definidas (comparações, funções matemáticas, lógicas, ...) ou definidas pelo usuário que recebem um ou mais argumentos e retornam verdadeiro quando estes satisfazem o predicado.

Tabela 4 – Números e tipos de argumentos suportados pelos tipos de átomos

Tipo de átomo	Número de argumentos	Tipo de argumento
<i>Class</i>	1	i-object
<i>Object property</i>	2	i-object, i-object
<i>Data valued property</i>	2	i-object, d-object
<i>Same/Different</i>	2	i-object, i-object
<i>Built-in</i>	2 ou mais	d-object ou i-object
<i>Data range</i>	1	d-object

Os átomos podem se referir a dois tipos de argumentos:

- **I-Object:** que representam os indivíduos definidos na ontologia ou variáveis para indivíduos;
- **D-Object:** que representam valores constantes ou variáveis para valores;

⁵ Na Tabela 3 o termo indivíduo é empregado para representar: 1) um indivíduo específico definido na ontologia; ou, 2) variáveis para indivíduos, que podem ser quaisquer indivíduos definidos na ontologia. Ou seja, corresponde ao tipo I-Object.

⁶ O termo valor neste caso é uma livre tradução de *data value* e representa um dado de um tipo primitivo definido na ontologia (Ex: inteiro, real, *string*). Na Tabela 3, o termo valor é empregado para representar valores constantes ou variáveis para valores. Ou seja, corresponde ao tipo D-Object.

As variáveis são tratadas como quantificadores universais e possuem o escopo limitado à regra a qual pertencem. Apenas variáveis que ocorrem no antecedente podem ocorrer no consequente.

Embora as regras SWRL possam ser representadas em mais de um formato, o formato de leitura humano é adotado neste trabalho. Neste formato, a seta ($\rightarrow$) é usada para separar antecedente e consequente, o acento circunflexo (^) representa a conjunção entre os átomos e o sinal de interrogação (?) distingue as variáveis dos nomes de indivíduos. Na Figura 3 é apresentada uma regra SWRL, representada no formato humano, ressaltando suas partes e características.

Figura 3 – Regra SWRL especificada no formato humano e suas partes.

Semanticamente, o exemplo anterior ilustra que **se** o pai/mãe de um indivíduo possui um irmão, **então** este irmão é tio do indivíduo em questão. A regra possui três átomos (todos do mesmo tipo⁷) sendo dois no antecedente e um no consequente (cujos predicados são: `hasParent`, `hasBrother` e `hasUncle` respectivamente) e faz uso de três variáveis para indivíduos: `?x1`, `?x2` e `?x3`.

Cabe ressaltar que as regras são armazenadas na própria ontologia e o uso deste tipo de anotação permite a inferência de novos conhecimentos sobre indivíduos, uma vez que as regras correspondem a afirmações condicionais que, ao serem satisfeitas, agregam novas informações à base de conhecimento.

⁷ Neste caso, é suposto que o nome dos predicados são propriedades definidas em uma ontologia

2.5 Considerações Finais

As ontologias possibilitam representar, organizar e compartilhar o conhecimento em um domínio e a Web Semântica se estabelece como a plataforma que usa ontologias para disponibilizar conteúdo semântico na Web. As linguagens RDF, RDFS e OWL são destacadas como a base da Web Semântica, uma vez que são os padrões mais utilizados para a criação de ontologias na Web (e anotação de dados).

Além disso, o uso de regras SWRL para realizar anotações semânticas como asserções permite a criação de afirmações condicionais que possibilita a inferência de novos conhecimentos sobre indivíduos em ontologias.

3 Ferramentas e Interfaces de Regras

Neste capítulo, são apresentadas as principais ferramentas para gerenciamento de regras SWRL encontradas na literatura. Além das ferramentas de regras específicas para SWRL, ferramentas para gerenciamento de regras de propósito geral foram analisadas. Neste levantamento, não foram consideradas as ferramentas denominadas de anotações para outros fins, apenas ferramentas com suporte a regras.

3.1 Introdução

A Web Semântica renovou e aumentou o interesse em sistemas baseados em regras e seu desenvolvimento (Zacharias, 2008). Na medida em que sistemas de regras, conhecidos por *Rule Based Systems* (RBS), compartilham características e funcionalidades, sua investigação pode revelar boas práticas e direcionar o seu desenvolvimento.

Nos dias atuais, muitos sistemas empresariais incluem um módulo ou utilizam um programa para escrever regras, estes programas correspondem a um tipo especial de RBS, que trabalham com regras de negócio, denominado *Business Rule System* (BRS). Neste caso, as regras podem ser classificadas em três categorias (Braye et al., 2006):

- **Regras dedutivas:** corresponde a regras que derivam novas informações, semelhante ao funcionamento das regras SWRL;
- **Regras de restrição:** correspondem a declarações que devem ser sempre válidas e são destinadas para modelagem de regras de negócio.
- **Regras de processo:** utilizadas para modelar as atividades de negócio, fazendo com que, ao serem satisfeitas, realizem determinadas ações no sistema. São também conhecidas por **regras de produção**.

Embora o uso da linguagem SWRL se restrinja à categoria das regras dedutivas, as regras das demais categorias possuem sempre a mesma estrutura condicional “SE...ENTÃO”. Isto ocorre pelo fato de as regras, em geral, serem instruções declarativas que indicam apenas “o que” e não como as coisas devem acontecer.

Durante a revisão das ferramentas RBS o principal foco de observação foram as interfaces e abordagens de interação empregadas no trabalho com regras. De um modo geral, os recursos encontrados são (Braye et al., 2006; Zacharias, 2008):

- Editor de texto;
- Ambiente de desenvolvimento integrado (*Integrated Development Environment* - IDE);
- Formulários;
- Editor gráfico;
- Diagramas UML;
- Planilhas;
- Árvore de decisão;
- Linguagem natural;
- Regras em conjunto com ontologias;
- Extração automática a partir de um conjunto de dados;

3.2 Ferramentas de Regras SWRL

3.2.1 SWRL Tab

A ferramenta SWRL Tab (O'Connor, Musen e Das, 2009) é um ambiente de desenvolvimento de regras SWRL, desenvolvido como um *plug-in* para o editor de ontologias Protégé. O Protégé é a principal tecnologia de uso livre para o desenvolvimento e gerenciamento de ontologias (Knublauch et al., 2004; Rubin, Noy e Musen, 2007). De um modo geral, a SWRL Tab possui diversos componentes de software que além de oferecer suporte à edição e execução de regras, disponibilizam um conjunto de bibliotecas para o trabalho com regras.

Na Figura 4 é ilustrado o editor de regras da ferramenta SWRL Tab, contendo uma listagem com as regras no formato tabular, no qual é possível editá-las diretamente (como ilustrado na imagem) ou em um editor textual à parte. A ferramenta realiza a identificação dos erros sintáticos conforme a especificação SWRL (Horrocks et al., 2004), valida o correto uso dos termos da ontologia, possui auto completar durante a composição e não oferece recursos relacionados à organização das regras.

Figura 4 – Editor de regras da ferramenta SWRL Tab⁸.

Além do editor de regras, um conjunto de *Application Programming Interfaces* (APIs) enriquece o conjunto de componentes disponíveis na ferramenta SWRL Tab. Estas APIs são desenvolvidas em Java e compreendem:

- **SWRL Rule Engine API:** utilizada para criar instâncias de máquinas de inferência para regras. Como padrão utiliza a máquina de inferência *Java Expert System Shell* (JESS) (Friedman-Hill, 2008);
- **SQWRL Query API:** permite que as regras possam ser acessadas por meio de *Java Database Connectivity* (JDBC), utilizando a linguagem de consulta *Semantic Query-Enhanced Web Rule Language* (SQWRL). SQWRL é uma linguagem de consulta de regras que utiliza a sintaxe SWRL juntamente com as operações *Structured Query Language* (SQL) para obtenção de conhecimento de uma base OWL (O'Connor e Das, 2009);
- **SWRL Factory:** utilizada para a manipulação das regras, incluindo a criação, edição e exclusão das regras. Esta API pode ser utilizada por desenvolvedores que desejam trabalhar com regras em suas aplicações. As regras são armazenadas como indivíduos das classes disponíveis na ontologia SWRL⁹. Todas as classes disponíveis na

⁸ Imagem disponível em <http://protege.cim3.net/file/work/images/SWRL/SWRLTab.png>

⁹ <http://www.daml.org/rules/proposal/swrl.owl>

ontologia possuem uma classe Java correspondente, permitindo assim realizar o mapeamento diretamente;

- **SWRL Built-in Bridge:** provê mecanismos para definição de implementações Java para a criação de *Built-ins*, garantindo que estes sejam executados corretamente pela máquina de inferência. Além disso, contém os *Built-ins* definidos na especificação SWRL (Horrocks et al., 2004);
- **SWRL Bridge:** define os padrões para definição de máquinas de inferência para regras e padrões para a criação dos *Built-ins*;

3.2.2 Axiomé

A ferramenta Axiomé (Hassanpour, O'Connor e Das, 2009) também é um *plug-in* da aplicação Protégé e faz uso das APIs disponibilizadas pela ferramenta SWRL Tab. Sob a perspectiva do usuário, o Axiomé amplia a quantidade de recursos voltados à organização e visualização das regras, além de disponibilizar a edição das regras por meio de *templates* gerados com base na estrutura das regras. As principais técnicas/interfaces que a ferramenta agrega são:

- Agrupamento automático das regras;
- Representação visual das regras;
- Paráfrases em regras (uma explicação textual da regra gerada dinamicamente);
- *Template* para edição e aquisição de regras;
- Gráficos de visualização e dependências das regras;

Os métodos de agrupamento, representação visual, paráfrases e os *templates* para criação das regras são definidos a partir de uma expressão canônica, que é obtida da análise léxica da regra combinada com a reordenação dos átomos (conforme seus tipos) e da aplicação de algumas heurísticas.

Na Figura 5 são ilustradas as interfaces utilizadas para representação visual (A), formulário de edição (B) e representação textual (C) de uma regra, além de exemplificar a expressão canônica (D) gerada e utilizada para agrupar a regra.

Figura 5 – Ferramenta Axiomé: (A) representação visual; (B) formulário de edição; (C) representação textual; (D) expressão gerada para o agrupamento de uma regra.

A representação visual utiliza retângulos para representar os átomos e os organiza visualmente de modo que o antecedente é apresentado na esquerda e o conseqüente na direita, separados por uma seta. A disposição visual mostra a ligação entre os átomos, que por sua vez tem seu conteúdo mapeado para frases em inglês.

A descrição textual da regra é utilizada para a geração do formulário de edição das regras, uma vez que os valores dos argumentos são mapeados para caixas de texto e os predicados para caixas de seleção (*combo boxes*). Os grupos são gerados automaticamente a partir da estrutura das regras. A estrutura em árvore que representa cada regra é mapeada para uma expressão canônica que indica a quantidade e os tipos de átomos utilizados. As regras, que apresentam a mesma expressão são classificadas no mesmo grupo.

Além destes recursos, está disponível uma visualização gráfica do conjunto de regras, na qual cada regra é representada como um vértice de um grafo e as arestas ligam duas regras indicando a dependência entre tais regras. Além disso, a ferramenta disponibiliza um mecanismo de busca de regras.

Do ponto de vista da usabilidade, a ferramenta Axiomé melhora significativamente a interação do usuário com as regras, contudo, alguns problemas dessa natureza ainda podem ser destacados:

- O formulário de edição não permite modificar a estrutura da regra, por exemplo, não é possível criar uma regra que possua uma estrutura distinta das demais;
- O agrupamento por meio da estrutura da regra não garante que os grupos possuam regras que tenham conceitos em comum. Além disso, esta abordagem faz com que alguns grupos tenham muitas regras, enquanto outros tenham apenas uma regra;
- O sistema de busca não permite refinar as buscas e não apresenta os resultados de forma clara ao usuário. Não é possível, por exemplo, buscar por um determinado tipo de átomo ou em apenas uma parte da regra.

Atualmente, a ferramenta está sendo migrada para a versão Web do Protégé (Tudorache, Vendetti, e Noy, 2008) e recursos semânticos estão sendo agregados à ferramenta (Hassanpour, O'Connor e Das, 2010).

3.2.3 ACE View

A ferramenta ACE View (Kaljurand, 2008) é um editor de regras e ontologias que utiliza o *Attempto Controlled English* (ACE) que pode ser usado para criar, visualizar e editar axiomas OWL e regras SWRL. O ACE é uma linguagem natural controlada, formada por um subconjunto da língua inglesa, destinada à representação do conhecimento. Embora se assemelhe muito à linguagem natural escrita, é uma linguagem formal que permite dois tipos de mapeamento: $ACE \rightarrow OWL/SWRL$ e $OWL \rightarrow ACE$.

Também desenvolvida como um *plug-in* do editor de ontologias Protégé, o ACE View utiliza um paradigma de interação diferente das demais ferramentas de regras SWRL ao utilizar apenas a linguagem natural para permitir a criação de regras e conceitos da ontologia. As sentenças descritas em ACE correspondem a axiomas OWL ou regras SWRL e as palavras correspondem a conceitos e propriedades da linguagem OWL.

O suporte à linguagem SWRL é limitado, pois, as regras previamente definidas em SWRL não podem ser convertidas em texto ACE. Na Figura 6 é apresentada a tela de gerenciamento da base de conhecimento (axiomas OWL e regras SWRL), na qual uma regra definida em linguagem natural foi convertida para a representação SWRL.

Figura 6 – Ferramenta ACE View na criação de uma regra SWRL

A ferramenta ACE View introduz um novo paradigma para a criação e visualização de OWL/SWRL ao eliminar o uso de formulários e editores sintáticos e permitindo que os desenvolvedores trabalhem com uma linguagem natural. Todavia, para utilizar a ferramenta ACE View é necessário conhecimento prévio da linguagem ACE.

3.3 Ferramentas de Regras de Negócio

As ferramentas utilizadas na modelagem de regras de negócio encontram-se num estágio mais consolidado se comparadas às ferramentas SWRL e sua investigação contribuiu no levantamento de requisitos, identificação das interfaces e definição das contribuições deste trabalho. A análise destas ferramentas foi conduzida a partir da revisão desenvolvida por Braye et. al. (2006). A seguir, são descritas brevemente as ferramentas analisadas.

- **BRS Rule Track** é uma ferramenta que utiliza a linguagem BRS RuleSpeak, na qual as regras são escritas em linguagem natural. Permite gerenciar os termos (conceitos) interativamente. Além disso, possui duas perspectivas de trabalho: perspectiva de negócio, destinado aos usuários finais e perspectiva tecnológica, destinado a profissionais técnicos em informática;
- **CLIPS**, acrônimo para *C Language Integrated Production System*, é uma ferramenta de domínio público amplamente utilizada na indústria e na academia. Embora CLIPS seja um sistema especialista e não uma ferramenta específica para regras de negócio, ela também pode ser utilizada para este fim, permitindo a definição de fatos, regras, funções e conceitos orientados a objetos. Existem ainda outras ferramentas para a criação de regras em CLIPS como CLIPSTab, um *plug-in* para o Protégé, e Clips Editor, disponível em <http://clipsrules.sourceforge.net/>;
- **Drools** é uma ferramenta integrada ao JBoos que utiliza uma linguagem de regras própria e introduz uma nova plataforma de integração de regras, na qual integra regras, fluxo de trabalho e processamento de eventos. Drools Guvnor é o repositório centralizado da base de conhecimento que possui interfaces Web, editores e ferramentas adequadas para gerenciar um grande número de regras; Drools Expert consiste na máquina de inferência de regras; Drools Flow foi renomeado para jBPM e consiste em uma plataforma para descrever fluxos de trabalhos e processos; Drools Fusion é responsável pelo processamento de eventos e questões temporais; Drools Planner é o módulo responsável para a realização de planejamento automatizado. Disponível em <http://jboss.org/drools.html>, esta ferramenta funciona como *plug-in* do Eclipse;
- **Ferramentas ORM** correspondem a um conjunto de ferramentas distintas que suportam a linguagem *Object Role Modeling* (ORM). A linguagem ORM é orientada a fatos permitindo expressar, transformar e buscar informações em nível conceitual por meio de notação gráfica, o que permite a criação de regras como diagramas. Ferramentas comerciais: Microsoft's Visio for Enterprise Architects e CaseTalk. Ferramentas de uso livre: VisioModeler, Infagon, NORMA e DOGMAModeler.
- **Fico Blaze Advisor** utiliza a linguagem *Structured Rule Language* (SRL) para criação de regras e permite que especialistas em negócios trabalhem em conjunto com desenvolvedores técnicos. Provê um ambiente integrado para criação, desenvolvimento e teste de regras de negócio. Disponível em <http://www.fico.com>;

- **IBM ILOG** é uma ferramenta de regras de negócio que utiliza uma linguagem de regras própria. Engloba um conjunto de componentes de software que juntos consistem em um ambiente integrado para o desenvolvimento de regras, além de permitir que especialistas em negócio trabalhem em conjunto com desenvolvedores técnicos. Disponível em <http://www.ibm.com/software/websphere/ilog/>;
- **LibRT Rule Management System (RMS)** permite que especialistas em negócio mantenham e gerenciem regras em diferentes formalismos e formatos de representação do conhecimento. Disponível em <http://www.librt.com/>;
- **OpenLexicon** é uma ferramenta de código aberto que além de suportar regras de negócio permite o gerenciamento de processos. Utiliza uma linguagem e um repositório de regras próprio. Disponível em: <http://www.openlexicon.org/>;
- **Oracle Business Rules** é uma ferramenta de regras de negócio que permite a criação de regras utilizando declarações próximas à linguagem natural. Disponível em <http://www.oracle.com/technetwork/middleware/business-rules/overview/>;
- **RuleXpress** utiliza a linguagem de regras BRS RuleSpeak e é semelhante à ferramenta BRS Rule Track. Disponível em <http://www.rulexpress.com/>;
- **SAP NetWeaver Business Rules Management** ou **QuickRules Editor** é uma ferramenta que permite o gerenciamento de várias fases do ciclo de vida de um sistema de regras: desenvolvimento, execução, modificação e otimização. Disponível em <http://www.sap.com/platform/netweaver/components/brm/>;
- **Translator** consiste em uma sigla para *TRANSlator from LAnguage TO Rules* é uma ferramenta de uso livre disponível em uma aplicação *Java Web Start*. Embora não seja uma ferramenta específica para regras de negócio, tem sido empregada para este fim devido à sua simplicidade para não especialistas. A ferramenta utiliza a linguagem ACE, e realiza a conversão para *Rule Markup Language* (RuleML), uma linguagem de regras da Web Semântica. Disponível em <http://ruleml.org/translator/>;
- **Visual Rules** é uma plataforma para gerenciamento de regras de negócio por meio de gráficos. Disponível em <http://www.visual-rules.com/>;

3.4 Interfaces

O levantamento e estudo das ferramentas (seção anterior) tiveram como principal finalidade a identificação das principais interfaces e recursos empregados no gerenciamento

de regras. A seguir, essas interfaces e recursos são listados e explicados. Eles foram utilizados para nortear as contribuições deste trabalho e serviram de instrumento de captação de requisitos nas reuniões com os usuários.

3.4.1 Tabelas de decisão

Presente na maioria das ferramentas de regras analisadas, as tabelas de decisão são compostas por linhas e colunas, como o próprio nome sugere. Elas utilizam o formato tabular para descrever ou apresentar todas as situações que podem ocorrer em um determinado cenário e quais ações devem ser executadas em cada caso. Esse tipo de interface é adequado para visualizar e gerenciar grandes conjuntos de regras com estruturas e conceitos semelhantes.

Normalmente, em uma tabela de decisão, cada linha corresponde a uma regra e cada coluna corresponde a uma condição ou ação que as regras possuem, de modo que, cada célula contém o valor ou expressão desta condição para uma regra em particular. Todavia, por se tratar de uma representação tabular as linhas e colunas podem ser alternadas. Na Figura 7 é apresentado um exemplo de uma tabela de decisão contendo duas regras.

	State	Age of the customer		Rental Rejected
		Minimum Age	Maximum Age	
1	New York	18	21	true
2	New York	21	25	false
3				

Figura 7 – Tabela de decisão em formato de planilha (IBM ILOG)

As ferramentas que possuem esta interface são: Drools; FICO Blaze Advisor; RuleXpress; IBM ILOG; LibRT Rule Management System; Oracle Business Rules; Sap NetWeaver BRM; e Visual Rules.

3.4.2 Árvores de decisão

As árvores de decisão são semelhantes às tabelas de decisão, mas são compostas por nós ao invés de linhas e colunas. É uma alternativa adequada para gerenciar grandes conjuntos de regras similares, especialmente quando as regras não são simétricas. Em uma árvore de

decisão, cada regra é representada pelo conjunto de nós que vai da raiz da árvore até uma folha. Assim, se uma árvore possui cinco folhas, isso significa que a mesma representa cinco regras.

Esta interface pode ser representada graficamente de maneiras distintas: os nós podem ser representados por elipses, retângulos ou outras formas geométricas; os rótulos e condições podem ser apresentados dentro ou fora de um nó ou nas arestas que ligam os nós; e uma árvore pode ser apresentada na posição horizontal ou vertical, além de ser possível empregar outras estruturas de representação hierárquica.

Embora árvores de decisão sejam frequentemente utilizadas para visualização de conjuntos de regras, esta estrutura pode ser utilizada para aquisição de novas regras. Seu uso é enriquecido quando são combinadas com mecanismos de interação como zoom, navegação e busca. Um exemplo de árvore de decisão com cinco regras é apresentado na Figura 8.

Figura 8 – Árvore de decisão utilizando a estrutura de fluxograma (IBM ILOG)

As ferramentas que possuem esta interface são: FICO Blaze Advisor; IBM ILOG; e LibRT Rule Management System.

3.4.3 Diagrama gráfico

Regras ou conjunto de regras podem ser representados por meio de diagramas que permitam que os usuários possam visualizar e criar novas regras. Quando um diagrama é utilizado para representar uma regra específica, formas geométricas e estilos representam a sintaxe e semântica da linguagem empregada. Por outro lado, ao representar um conjunto de regras, os diagramas enfatizam características específicas ou o relacionamento entre as regras. Na Figura 9, é apresentado um diagrama que mostra dependência entre regras, no qual cada círculo representa uma regra e as setas indicam as dependências.

Os diagramas podem ser enriquecidos com cores, textos, formas, imagens, símbolos e traços, além disso, recursos de interação podem auxiliar na manipulação dos diagramas. Um tipo especial de diagrama, chamado *rule flow*, combina árvore de decisão com imagens, cores e símbolos (Figura 10).

As ferramentas que possuem esta interface são: Axiomé; Ferramentas ORM; IBM ILOG (*Graphical Business Object Model Editor*); e, Visual Rules.

Figura 9 – Diagrama de dependência entre as regras (Axiomé)

Figura 10 – Rule flow utilizado para criar e visualizar regras (Visual Rules)

3.4.4 Editor de texto

Editores de texto são utilizados normalmente por desenvolvedores com conhecimento computacional, uma vez que o domínio da linguagem de regras utilizada se faz necessário. Embora alguns editores se assemelhem a um bloco de notas, outros são mais sofisticados como IDEs possuindo recursos como auto completar, *highlight* da sintaxe, auto indentação, ferramentas de erros e avisos, *debug* automatizado e testes. Na Figura 11 é apresentado um exemplo de editor de texto com *highlight*, apresentando duas regras.

Figura 11 – IDE Eclipse com recursos de *highlight* (Drools)

As ferramentas que possuem esta interface são: Drools (editor do Eclipse); RuleXpress; e, SWRL Tab.

3.4.5 Formulários e *templates*

Formulários e *templates* são interfaces utilizadas para a composição de regras nas quais o usuário preenche os campos disponíveis. Enquanto os formulários são estruturas mais genéricas, os *templates* são disponibilizados especificamente para um grupo de regras

estruturalmente semelhantes. O uso destas interfaces torna possível abstrair detalhes da sintaxe e eventuais complexidades das regras.

Algumas ferramentas permitem que os desenvolvedores técnicos possam gerar os *templates* para os usuários especialistas no domínio, enquanto outras os geram automaticamente conforme a estrutura das regras disponíveis. Um *template* de exemplo é apresentado na Figura 12, neste caso apenas algumas partes da regra em edição podem ser modificadas.

Rule:

If:
Consider all of the following conditions:

☐ The total protection needed of theInsured is between and

[< Select new condition >](#)

Then:

☐ Set the incentive plan of theInsured to

Figura 12 – Template Web para a criação de uma regra (FICO Blaze Advisor)

As ferramentas que possuem esta interface são: Axiomé; Drools; FICO Blaze Advisor; IBM ILOG; OpenLexicon; Oracle Business Rules; e SAP NetWeaver BRM.

3.4.6 Interface interativa (*shell*)

Shell é um software que possui como interface com usuário um console de texto que aceita e executa comandos. A interação com o usuário é por meio de linhas de comandos, uma interface simples e direta, o que a torna mais apropriada a desenvolvedores técnicos. Das ferramentas analisadas, apenas a CLIPS, apresentada na Figura 13, suporta este tipo de interface.

```

Dialog Window
CLIPS (V6.24 06/15/06)
CLIPS> (defrule red-light
(light red)
=>
(printout t "Stop" crlf))
CLIPS> (defrule green-light
(light green)
=>
(printout t "Go" crlf))
CLIPS> (assert (light green))
<Fact-0>
CLIPS> (run)
Go
CLIPS>

```

Figura 13 – Interface interativa (CLIPS)

3.4.7 Linguagem natural

Do ponto de vista do usuário, a linguagem natural é a maneira mais simples de criar e visualizar regras, contudo, da perspectiva tecnológica, este tipo de interface apresenta um grande conjunto de desafios como, por exemplo, a resolução de ambiguidades. Isso acarreta no fato deste tipo de interface ser mais encontrada na visualização do que na composição de regras.

Todas as ferramentas observadas que oferecem suporte a linguagem natural utilizam o inglês, sendo que em alguns casos, os textos são enriquecidos com *highlights* evidenciando conceitos, funções da linguagem ou ressaltando partes da regra. O suporte da linguagem de regras com relação a linguagem natural influencia diretamente na adoção ou não deste tipo de interface pela ferramenta. Na Figura 14, é apresentado um conjunto de regras em linguagem natural combinadas com operadores aritméticos onde cores são usadas para destacar alguns termos.

```

If at least 2 children satisfy age < 8
then set discount to 0.25.

If Product's ID does not start with "SPX"
then the PromotionStatus of the Product is False.

If the name of the customer is unknown
then print("Please enter your name").

If order's purchaseDate is earlier than 'January 1, 2002'
then print("Your purchase is no longer eligible for return").

SeniorMale is any customer such that (age > 65 and gender is Male).

If the max of the weight of every bag is less than 100
then the status of the cargo is "OK"
  
```

Figura 14 – Visualização de regras em linguagem natural com *highlight* (FICO Blaze Advisor)

As ferramentas que possuem esta interface são: ACE View, Axiomé; BRS Rule Track; FICO Blazer Advisor; ORM Tools; RuleXpress; e, Translator.

3.5 Outros Recursos

Além das interfaces, foram identificados recursos (componentes de interface e funcionalidades) disponíveis nas ferramentas que enriquecem a interação do usuário durante o processo de criação e visualização das regras.

3.5.1 Nome, descrição e partes de regras

A maioria das linguagens de regras, e consequentemente das ferramentas, permitem que as regras possam ser nomeadas e associadas a uma descrição textual, fazendo com que os desenvolvedores possam gerenciá-las mais facilmente. O uso de nomes e descrições auxilia na organização e no processo de busca. O gerenciamento de grandes conjuntos de regras, pode se tornar inviável sem uma identificação adequada das mesmas.

Outra possibilidade neste sentido é o gerenciamento e divisão das regras em partes, que podem até mesmo ter nomes, o que torna seu gerenciamento mais fácil e prático. Estas partes podem ser denominadas de átomos, fragmentos ou segmentos de regras. O uso de partes das regras se mostra mais eficaz no processo de composição das mesmas, pois permite que uma regra possa ser criada a partir de sub-regras distintas.

3.5.2 Representações visuais

Algumas ferramentas fazem uso de representações visuais para abstrair regras ocultando detalhes sintáticos da linguagem ou explicitando informações. Essas representações visuais se assemelham aos diagramas, contudo são utilizadas para representar regras individuais. Para tanto, fazem uso de cores, formas, traços e estilos que variam conforme a ferramenta e a linguagem de regra utilizada. As representações podem utilizar as descrições textuais ou ressaltar partes da regra tornando-a mais significativa para o usuário.

Na Figura 5.A, é apresentada a representação visual disponível na ferramenta Axiomé, na qual cada átomo da regra é representado por um retângulo. Esses retângulos são organizados hierarquicamente conforme os argumentos presentes nos predicados. Neste caso, a representação está sendo empregada na visualização, contudo, esse tipo de recurso também pode ser empregado na composição de regras.

3.5.3 Grupos e categorias

O agrupamento de regras é comum na maioria das ferramentas analisadas e contribui para a organização e classificação das regras. Em algumas ferramentas, o agrupamento ocorre de forma manual (definido pelo criador das regras) e, em outras, de modo automático (definido por algum algoritmo). Os agrupamentos podem permitir o uso de subgrupos (estrutura hierárquica) e também a categorização em mais de um grupo (utilizando *tags*).

A criação de grupos manuais possibilita maior controle e organização sob a perspectiva do desenvolvedor, porém, ao mesmo tempo deixa toda a responsabilidade de organização sobre este, que pode aplicar critérios subjetivos ou contraditórios. Por outro lado, a geração de grupos automáticos, que utiliza critérios bem definidos, está submetida a gerar agrupamentos confusos ou sem utilidade prática. Os algoritmos empregados na geração de grupos usam como argumentos as características, estruturas e partes das regras fazendo com que a menor das alterações em uma regra possa implicar em sua reclassificação em outro grupo.

3.5.4 Componentes de sugestão

A sugestão de termos contribui e melhora o processo de composição de regras ao sugerir termos, palavras reservadas, operadores e funções da linguagem para compor as regras. São utilizadas caixas de seleção e menus suspensos que apresentam o conteúdo organizado em grupos, de modo hierárquico ou em uma simples lista.

As sugestões são geradas a partir do conteúdo previamente criado ou da expressão que está sendo digitada (auto completar), nesse caso contribuindo para a diminuição de erros de digitação.

3.5.5 Ícones e símbolos

O uso de ícones e símbolos é comum as ferramentas de regras e são aplicados em conjunto com outros recursos como, por exemplo, as representações visuais. Esse recurso pode estar acompanhado com textos e efeitos visuais e seu uso é empregado para dois fins:

- Representar os recursos da ferramenta;
- Representar partes ou características das regras;

3.6 Considerações Finais

As ferramentas analisadas que suportam a linguagem SWRL, são *plug-ins* do editor de ontologias Protégé e não possuem uma versão Web. Se comparadas com as demais ferramentas apresentadas possuem poucos recursos e funcionalidades, tornando difícil o processo de visualização e composição de regras, especialmente quando o conjunto se torna grande. Existe um grande número de técnicas e ferramentas relacionadas à criação de regras de negócio que podem apontar a direção para o aprimoramento e criação de novas técnicas relacionadas a regras SWRL (Rivolli, Orlando e Moreira, 2011).

Embora os avanços das ferramentas e os estudos relacionados à criação de regras de negócio ainda não estejam consolidados (Zacharias, 2008), o estágio atual de seu desenvolvimento está muito à frente se comparado a regras SWRL. A análise destas ferramentas mostra as distintas formas de interação do usuário com as regras, como por exemplo: tabelas e árvores de decisão, *templates* para edição de regras, linguagem natural, diagramas para visualização e editores integrados que utilizam técnicas de agrupamento, sugestão de termos, detecção e correção de erros.

Os resultados obtidos ilustram o cenário atual das ferramentas de regras, suas principais interfaces e funcionalidades o que permitiu direcionar o desenvolvimento do trabalho.

4 Aprimorando o uso de regras SWRL

4.1 Introdução

Neste capítulo, são apresentadas as soluções desenvolvidas para aprimorar o uso de regras SWRL. Os resultados foram obtidos por meio de um processo iterativo e contou com a colaboração dos desenvolvedores de regras SWRL do grupo BMIR (*Stanford University School of Medicine*) e outros estudantes do ICMC envolvidos com o projeto.

As ferramentas, interfaces e recursos apresentados no Capítulo 3 foram apresentadas para os desenvolvedores de regras SWRL do grupo BMIR, que por sua vez ajudaram na definição dos requisitos e recursos necessários para o desenvolvimento da ferramenta SWRL Editor. A participação dos desenvolvedores neste processo ocorreu por meio de videoconferências e foi de fundamental importância para esta fase do projeto.

Tais requisitos compreendem:

- Árvore de decisão;
- Editor de texto;
- Formulários;
- Linguagem natural;
- Representações visuais;
- Agrupamento;
- Componentes de sugestão;
- Ícones e símbolos;
- Mecanismo de busca;
- Customização;

Embora alguns destes itens estejam presentes nas ferramentas SWRL Tab e Axiomé, tais recursos foram aprimorados, além disso, estas ferramentas não estão disponíveis na Web. As soluções desenvolvidas e apresentadas neste capítulo compreendem:

1. Desenvolvimento de uma API para representar o modelo lógico das regras SWRL (Seção 4.2);
2. Elaboração de duas representações visuais para representar regras SWRL (Seção 4.4);

3. Desenvolvimento de uma técnica para identificar o fator de similaridade entre regras SWRL, utilizadas pelas técnicas de agrupamento e identificação de erros (Seção 4.5);
4. Desenvolvimento de uma técnica para agrupar estas regras a partir dos fatores de similaridade (Seção 4.6);
5. Identificação, classificação e desenvolvimento de um algoritmo para detecção de erros nas regras SWRL (Seção 4.7);
6. Desenvolvimento de uma ferramenta Web, SWRL Editor, que integra as representações visuais elaboradas e as técnicas desenvolvidas, juntamente com um mecanismo de busca (Seção 4.8);

A colaboração entre nosso grupo e o *BMIR* levou ao desenvolvimento da ferramenta como um *plug-in* para a versão Web do editor de ontologias Protégé (Web Protégé), que é uma ferramenta gratuita, extensível e de código aberto, desenvolvida pelo centro de informática biomédica da *Stanford University*.

As técnicas de análise de similaridade, agrupamento e detecção de erros foram propostas para auxiliar o processo de composição e visualização das regras SWRL e consequentemente melhorar a interação com o usuário.

As regras utilizadas como exemplo neste capítulo estão disponíveis na ontologia *Family History* (Peace e Brennan, 2008), que define possíveis relações entre membros de uma família. Esta ontologia contém um grande conjunto de regras, porém estas possuem em média poucos átomos e são simples, o que a tornou apropriada para ilustrar os exemplos.

4.2 Representação Lógica das Regras

A ferramenta Protégé está num momento de transição, existem as versões 3.x, baseadas numa API antiga com suporte a OWL 1.1, e as versões 4.x, baseadas na OWL-API (Horridge e Bechhofer, 2009) com suporte a OWL 2.0. Para garantir a portabilidade entre as versões foi desenvolvida uma API, denominada de SWRL Editor API, que trabalha com as regras em um nível mais alto de abstração¹⁰, tornando o trabalho mais fácil e independente de uma plataforma específica.

¹⁰ Diferentemente da sintaxe e representação apresentadas na Seção 2.4, as regras SWRL e suas partes são representadas internamente pelo editor de ontologias Protégé como indivíduos que pertencem a uma ou mais classes e possuem propriedades que os relacionam, como pode ser visto na especificação W3C (Horrocks et al., 2004), nas seções *XML Concrete Syntax* e *RDF Concrete Syntax*.

Nessa API, as regras são representadas do modo mais próximo possível ao modelo lógico apresentado na Seção 2.4. Na Figura 15, são apresentadas as principais classes e seus relacionamentos utilizados na representação das regras em SWRL.

Figura 15 – Principais classes da SWRL Editor API

Na Figura 16, é apresentado um diagrama de objetos que demonstra como são representados pela SWRL Editor API os átomos, predicados e argumentos da regra:

hasParent(?x1,?x2) ^ hasBrother(?x2,?x3) → hasUncle(?x1,?x3)

Essa regra possui três átomos mapeados nos objetos a1, a2 e a3, sendo que dois estão no antecedente e um no consequente; três predicados do tipo *Object property* mapeados nos objetos p1, p2 e p3 e relacionados aos átomos a1, a2 e a3 respectivamente; e, por fim, três argumentos mapeados nos objetos arg1, arg2 e arg3 todos do tipo *Individual*.

A SWRL Editor API possui um módulo de integração com as API do Protégé versões 3 (OWL versão 1) e 4 (OWL versão 2) para obtenção dos conceitos definidos na ontologia e suas regras, o que permitiu o desenvolvimento de um *plug-in* para esta ferramenta. A partir desse modelo, foram desenvolvidos algoritmos para gerar as representações visuais, analisar a similaridade entre regras, fazer agrupamentos e detectar erros em regras SWRL.

Figura 16 – Diagrama de objetos ilustrando como a SWRL Editor API representa os átomos, predicados e argumentos de uma regra

4.3 Protótipos de Interface

Durante o desenvolvimento das soluções, assim como da ferramenta SWRL Editor, foram elaborados protótipos e versões alternativas que permitiram aos envolvidos explorar cenários, avaliar os protótipos e propor melhorias. Por meio de reuniões pela Web com os desenvolvedores de regras (do BMIR), diversas questões foram definidas e as principais dúvidas esclarecidas.

Com relação às representações visuais, foram elaboradas versões alternativas, apresentadas aos desenvolvedores de regras, juntamente com as vantagens e desvantagens percebidas em cada uma delas. Os envolvidos puderam analisar e debater até chegar a um consenso sobre quais modelos de representação eram mais adequados para as regras SWRL. No Apêndice A, são ilustrados alguns exemplos elaborados e utilizados nas reuniões.

Foram utilizados protótipos de baixa fidelidade, por ser um mecanismo rápido e por atingir os objetivos esperados, para a concepção e validação da ferramenta SWRL Editor. Na Figura 17 é apresentada uma das telas propostas para a edição de regras, neste caso foram incluídas marcações e notas para facilitar o debate, que abordou os recursos previstos e fatores como estilos, estruturas e utilização da ferramenta. No Apêndice B são apresentados os protótipos elaborados.

Figura 17 – Protótipo da ferramenta SWRL Editor (tela de composição das regras)

4.4 Representações Visuais de Regras

Foram elaboradas duas representações visuais para abstrair as regras SWRL denominadas de SWRL *Highlight* e Visualização Hierárquica. Essas representações foram inicialmente usadas para aprimorar apenas a visualização (e compreensão) das regras SWRL, porém posteriormente foram usadas também no processo de composição (e escrita) de regras.

4.4.1 SWRL *Highlight*

A representação visual SWRL *highlight* exibe a regra sem ocultar os detalhes da sintaxe, contudo, agrega cores aos átomos e argumentos semelhantemente a recursos encontrados em alguns IDEs. Ao invés de utilizar a quebra automática no final da linha, cada átomo é apresentado em uma linha e o símbolo de implicação $\rightarrow$, que divide o antecedente do consequente, é disposto em uma linha separada, o que torna a regra mais legível. Na Figura 18, a regra `def_has_half_sister2` é apresentada no formato original (A) e na representação com *highlight* (B).

(A)

```

has_natural_father(?a, ?f) ^ has_natural_mother(?a, ?m) ^
has_natural_father(?b, ?d) ^ has_natural_mother(?b, ?m) ^ differentFrom(?f, ?d) ^
Person_Female(?b) -> has_half_sister(?a, ?b)

```

(B)

```

Person_Female(?b) ^
has_natural_father(?a, ?f) ^
has_natural_mother(?a, ?m) ^
has_natural_father(?b, ?d) ^
has_natural_mother(?b, ?m) ^
differentFrom(?f, ?d)
→
has_half_sister(?a, ?b)

```

Figura 18 – Comparativo entre (A) uma regra apresentada no formato original e (B) a representação visual SWRL *Highlight*

Na Tabela 5 são apresentadas as cores utilizadas na geração dos *highlights* e seus significados. A representação visual destaca os tipos dos átomos, variáveis e valores literais sem sobrecarregar a apresentação da regra. Algumas das cores utilizadas foram extraídas dos símbolos utilizados pelo editor de ontologias Protégé, enquanto outras foram extraídas do editor Eclipse (esta lista de cores é customizável através de folhas de estilo CSS). O uso de uma fonte de tamanho fixo (monoespaçada) foi adotada seguindo o padrão adotado nos editores de texto usados na programação, o que torna a leitura da regra mais organizada.

Tabela 5 – Cores utilizadas na representação SWRL *Highlight*

Cor	Exemplo	Descrição
Laranja	Person_Female	Representa predicados do tipo <i>Class</i> ¹¹
Azul	has_natural_father	Representa predicados do tipo <i>Object property</i>
Verde	has_current_age	Representa predicados do tipo <i>Datavalued property</i>
Vermelho	sqwrl:avg	Representa predicados do tipo <i>Built-in</i>
Cinza	differentFrom	Representa predicados do tipo <i>Same/different</i>
Preto em negrito	?a	Representa as variáveis
Roxo	32; "teste"	Representa os valores literais (Strings e números)

Embora seja possível representar mais detalhes como, por exemplo, distinguir os tipos de variáveis ou os tipos inteiros e strings, optou-se por não incluir tais detalhes para evitar

¹¹ Embora o Protégé utilize amarelo no símbolo que representa as classes, foi adotado para esta visualização o tom mais alaranjado pelo fato da cor amarela apresentar pouco contraste com fundos claros.

poluição visual, o que causaria sobrecarga para o usuário. O modelo adotado foi obtido após diversos testes e simulações.

4.4.2 Visualização Hierárquica

A representação denominada visualização hierárquica busca abstrair a sintaxe da regra apresentando os átomos agrupados hierarquicamente a partir dos átomos do tipo *Class*. Assim, ao mesmo tempo em que torna a regra mais clara para apresentação, enriquece com cores e símbolos o seu significado. Os símbolos empregados na ferramenta Protégé foram reutilizados, contudo, para representar os demais elementos (*Same/Different* e *Built-ins*) foram empregados retângulos com as cores utilizadas na representação SWRL *highlight*. Na Figura 19 são apresentados exemplos dos símbolos utilizados na visualização.

?b Person_Female	<i>Class</i>
has_natural_father(?a,?f)	<i>Object property</i>
has_current_age(?a,?b)	<i>Datavalued property</i>
?m same ?n ?m different ?n	<i>Same/Different</i>
?c > 18	<i>Built-ins</i>

Figura 19 – Símbolos utilizados na visualização hierárquica

O método de geração desta representação se assemelha, em alguns aspectos, ao utilizado pela ferramenta Axiomé (Hassanpour, O'Connor e Das, 2009), contudo, a representação inova ao utilizar cores distintas e símbolos na representação dos átomos e apresentar a regra na posição vertical ao invés da horizontal, organizando melhor a distribuição dos átomos. Na Figura 20, a regra **def_has_half_sister2** é apresentada utilizando a visualização hierárquica (A) e a representação utilizada pela ferramenta Axiomé (B).

Figura 20 – Comparativo entre (A) a representação visual hierárquica e (B) a representação utilizada pela ferramenta Axiomé

O método desenvolvido na geração da visualização utiliza o seguinte algoritmo:

Início

1. Exiba o marcador do antecedente (IF);
2. Insira todos os átomos do antecedente em uma lista;
3. Selecione os átomos do tipo *Class* e para cada átomo encontrado:
 - 3.1. Exiba a representação do átomo;
 - 3.2. Evidencie seu argumento;
 - 3.3. Selecione os átomos do tipo *Object property* e *Datavalued property* que possuem o argumento evidenciado como primeiro argumento e exiba-os a direita da representação do átomo do tipo *Class* correspondente;
 - 3.4. Retire todos os átomos exibidos da lista;
4. Exiba e retire da lista os átomos do tipo *Object property* e *Datavalued property* restantes;
5. Exiba e retire da lista os átomos do tipo *Same/Different*;
6. Selecione os átomos do tipo *Built-ins*, para cada átomo encontrado:
 - 6.1. Aplique os sinais e operadores adequados para representá-lo;
 - 6.2. Exiba a representação do átomo;
 - 6.3. Retire-o da lista;
7. Exiba e retire da lista os átomos restantes¹²;
8. Exiba o marcador do consequente (THEN);
9. Insira todos os átomos do consequente na lista;
10. Repita os passos de 3 a 7;

Fim.

¹² Se houver algum átomo restante na lista, este será do tipo *Data range*.

Embora esta representação oculte alguns detalhes da sintaxe SWRL e exiba os átomos organizadamente, seu uso se torna apropriado a desenvolvedores não técnicos em computação, apenas quando são utilizados na ontologia nomes apropriados para *labels*¹³ (evitando o uso dos *rdf:Ids* que têm regras de sintaxe mais restritas).

4.5 Análise de Similaridade

Mensurar um fator que determine o quanto duas regras são similares ou diferentes se mostrou uma opção capaz de auxiliar na visualização, organização e no processo de composição das regras. Ao utilizar este fator, as regras são organizadas em listas, grupos ou categorias de regras similares, podendo ser apresentadas de maneira significativa para os usuários. Na composição, é possível sugerir novos termos baseando-se em regras similares e utilizar a similaridade para identificação de erros, como o de regras duplicadas.

O primeiro passo desse algoritmo de similaridade consiste na elaboração de uma matriz de características, como ilustrado na Figura 21. As linhas correspondem às regras e as colunas correspondem aos átomos. Para cada linha/coluna é atribuído o número de ocorrências do átomo na regra.

	Átomo ₁	Átomo ₂	...	Átomo _N
Regra ₁	1	0	...	1
Regra ₂	1	1	...	1
...	...	...	...	...
Regra _N	1	1	...	0

Figura 21 – Modelo da matriz de característica utilizada na análise de similaridade

A partir da matriz de característica obtida, é possível determinar a distância entre duas regras P e Q quaisquer pela expressão:

$$d(P, Q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$

Onde: $P = [p_1, p_2, \dots, p_n]$ e $Q = [q_1, q_2, \dots, q_n]$ são os vetores de características das regras P e Q, respectivamente. A fórmula da distância Euclidiana, utilizada para o cálculo da

¹³ O uso de *labels* permite rotular os termos da ontologia com linguagem natural

similaridade, fornece a distância entre dois pontos em um plano n-dimensional (Salzberg, 1991). Com isso, quando duas regras possuem exatamente os mesmos átomos o resultado obtido pela aplicação da fórmula é zero e as regras são consideradas idênticas. O resultado indica a distância entre as regras, assim, quanto maior o valor obtido maior é a diferença entre as regras analisadas, e consequentemente, menor a similaridade entre elas.

Para ilustrar a aplicação do algoritmo, considere as regras **def_has_granddaughter**, **def_has_grandson** e **def_has_son2**, representadas por R1, R2 e R3 respectivamente (a matriz de características obtida é apresentada na Tabela 6):

- **R1:** $\text{has_grandparent}(?a, ?b) \wedge \text{Person_Female}(?a) \rightarrow \text{has_granddaughter}(?b, ?a)$
- **R2:** $\text{has_grandparent}(?a, ?b) \wedge \text{Person_Male}(?a) \rightarrow \text{has_grandson}(?b, ?a)$
- **R3:** $\text{has_natural_father}(?a, ?b) \wedge \text{Person_Male}(?a) \rightarrow \text{has_natural_son}(?b, ?a)$

Cujos átomos são:

- **A1:** $\text{has_grandparent}(?a, ?b)$
- **A2:** $\text{Person_Female}(?a)$
- **A3:** $\text{has_granddaughter}(?b, ?a)$
- **A4:** $\text{Person_Male}(?a)$
- **A5:** $\text{has_grandson}(?b, ?a)$
- **A6:** $\text{has_natural_father}(?a, ?b)$
- **A7:** $\text{has_natural_son}(?b, ?a)$

Tabela 6 – Matriz de características com os átomos das regras def_has_granddaughter, def_has_grandson e def_has_son2

	A1	A2	A3	A4	A5	A6	A7
R1	1	1	1	0	0	0	0
R2	1	0	0	1	1	0	0
R3	0	0	0	1	0	1	1

Aplicando a fórmula ao comparar as regras entre si, os seguintes valores foram obtidos:

- **d(R1, R2) = 2;**
- **d(R1, R3) = 2,44;**
- **d(R2, R3) = 2;**

Tais valores mostram, por exemplo, que a regra R1 é mais similar à regra R2, do que a regra R3. Isso porque R1 e R2 compartilham o átomo A1 enquanto que R1 e R3 não possuem

nada em comum. O mesmo acontece com R2 e R3 em comparação com R1 e R3, contudo o átomo compartilhado neste caso é o A4.

Utilizando a mesma técnica, mas aplicando algumas variações na matriz de características, outros cenários de uso podem ser explorados. Ao invés de utilizar a regra como um todo, pode ser utilizado apenas o antecedente ou consequente. Além disso, os argumentos podem ser descartados utilizando apenas os predicados. Desta forma, tais variações ocorrem em duas dimensões:

- **Dimensão 1 (linhas):** 1 – A regra toda, 2 – antecedente e 3 – consequente;
- **Dimensão 2 (colunas):** 1 – Predicado e 2 – átomos;

A escolha entre o uso do predicado ou átomo possui maior impacto na interpretação dos resultados do que as alterações da primeira dimensão. Enquanto a primeira dimensão restringe o número de átomos considerados na análise, ou em outras palavras, (1) o quanto as regras são similares; (2) o quanto os antecedentes das regras são similares; e, (3) o quanto os consequentes das regras são similares; a segunda dimensão causa maior impacto nos valores utilizados para a análise. Para ilustrar os efeitos que a escolha entre o uso do predicado ou átomo causa na análise, considere as regras abaixo onde R4 e R5¹⁴ representam a regra **def_has_nephew1** e R6 a regra **def_has_nephew2**:

- **R4:** `has_aunt(?a, ?b) ∧ Person_Male(?a) → has_nephew(?b, ?a)`
- **R5:** `has_aunt(?a, ?c) ∧ Person_Male(?a) → has_nephew(?c, ?a)`
- **R6:** `has_uncle(?a, ?b) ∧ Person_Male(?a) → has_nephew(?b, ?a)`

Cujos átomos são:

- **A4:** `Person_Male(?a)`
- **A8:** `has_aunt(?a, ?b)`
- **A9:** `has_nephew(?b, ?a)`
- **A10:** `has_aunt(?a, ?c)`
- **A11:** `has_nephew(?c, ?a)`
- **A12:** `has_uncle(?a, ?b)`

E, cujos predicados são:

¹⁴ As regras R4 e R5 são exatamente a mesma regra, mas com nomes de argumentos diferentes. Do ponto de vista lógico, estas regras são equivalentes.

- **P1:** has_aunt
- **P2:** Person_Male
- **P3:** has_nephew
- **P4:** has_uncle

A partir desse conjunto de regras, átomos e predicados a matriz de características obtida é apresentada na Tabela 7.

Tabela 7 – Matriz de características com os átomos e predicados das regras def_has_nephew1 (original e modificada) e def_has_nephew2

	Átomos						Predicados			
	A4	A8	A9	A10	A11	A12	P1	P2	P3	P4
R4	1	1	1				1	1	1	
R5	1			1	1		1	1	1	
R6	1		1			1		1	1	1

Aplicando a fórmula ao comparar as regras entre si, os valores obtidos utilizando apenas os átomos, predicados e ambos são apresentados na Tabela 8.

Tabela 8 – Valores obtidos na análise de similaridade utilizando átomos, predicados e a combinação de ambos

	Átomo	Predicado	Átomos e Predicados
d(R4, R5)	2	0	2
d(R4, R6)	1,41	1,41	2
d(R5, R6)	2	1,41	2,44

Enquanto no primeiro exemplo, o uso dos átomos se mostrou satisfatório, neste caso, o mesmo não ocorreu. Pelo fato de R4 e R5 serem a mesma regra, o grau de distância esperado deveria ser 0 e a mesma distância entre as regras R4 e R6, e R5 e R6 eram esperadas. Contudo, os resultados desejáveis ocorreram na utilização do predicado. Por fim, foi realizada uma análise utilizando átomos e predicados simultaneamente, o que não se mostrou satisfatório. Outras simulações foram realizadas, nas quais foram utilizados pesos diferentes para predicados e átomos, sem obtenção de êxito. A análise separada se mostrou a mais adequada, de modo que o uso dos átomos indica regras similares até mesmo nos nomes dos argumentos e o uso dos predicados revelam regras que se referem aos mesmos conceitos.

4.6 Agrupamento

O agrupamento automático das regras SWRL a partir de sua estrutura (Hassanpour, O'Connor e Das, 2009) não garante que os grupos gerados contenham regras cujos significados tenham ligação entre si, uma vez que o agrupamento é realizado apenas pela quantidade e tipos de átomos que as regras possuem (estrutura da regra), desconsiderando os valores destes átomos (significado da regra). Na busca de gerar grupos de regras significativos sob a perspectiva do usuário, o fator de similaridade foi considerado juntamente com a aplicação do método de agrupamento *K-means* (Jain, Murty e Flynn, 1999).

O *K-means* é classificado como um algoritmo divisível, fazendo com que os grupos sejam obtidos em partições simples, sem uma estrutura hierárquica. Isso faz com que possa ser aplicado em pequenos e grandes conjuntos de dados, mas precisa que o número de grupos seja pré-definido. O método não requer intervenção ou supervisão humana (não supervisionado) e os dados iniciais não precisam ser pré-classificados. Sua escolha foi motivada pela facilidade de implementação do algoritmo, pois o objetivo de seu emprego é validar o conceito de agrupamento por similaridade em substituição ao uso da estrutura da regra.

O algoritmo utilizado para o agrupamento das regras, a partir da similaridade, pode ser descrito em cinco passos:

Início

1. Selecionar k regras aleatórias, que serão utilizadas como centro dos grupos (centroides);
2. Calcular a similaridade entre cada regra e os centroides;
3. Agrupar cada regra com o centroide mais similar;
4. Calcular os novos centroides para os k grupos;
5. Voltar ao passo 2 até que haja convergência, ou seja, nenhum grupo sofra alterações e os centroides sejam mantidos;

Fim.

Por se tratar de um algoritmo não determinístico não é possível garantir que os grupos sejam preservados a cada aplicação do método, o que se mostrou um ponto negativo. Uma das formas de resolver esse problema é realizar a aplicação do método diversas vezes para encontrar a configuração de agrupamento que mais se repete e com isso persisti-la fisicamente, para que possa ser utilizada posteriormente. Embora essa alternativa seja viável,

a adição de novas regras ao conjunto faz com que seja necessária a reaplicação do algoritmo e reclassificação dos grupos.

Por outro lado, o algoritmo permite a formação de grupos contendo regras que possuem estruturas diferentes, mas significativas, o que torna o agrupamento mais rico quando comparado com a estratégia utilizada pela ferramenta Axiomé, o que é um ponto positivo.

Por fim, juntamente com o algoritmo de agrupamento, foi também criado um mecanismo de *plug-in* para permitir que outros algoritmos de agrupamento possam ser incorporados ao SWRL Editor. Isso facilitará a criação e teste de novas técnicas de agrupamento, ao permitir o reuso da interface gráfica do SWRL Editor no seu desenvolvimento.

4.7 Detecção de Erros

A identificação e correção de erros são um dos mais importantes requisitos apontados pelos desenvolvedores de regras (Zacharias, 2008). Desta maneira, integrar estes recursos na SWRL Editor API e na ferramenta SWRL Editor se mostrou essencial para aprimorar o processo de composição das regras.

A partir da especificação SWRL foi definida uma lista com os possíveis erros que os desenvolvedores de regras estão sujeitos. A lista vai além dos erros indicados na especificação, inclui alguns avisos e propõe uma classificação de escopo e tipo na organização dos erros identificados.

A classificação de escopo compreende:

1. Regra;
2. Átomo;
3. Argumento;

E, a classificação de tipo, compreende:

1. Aviso;
2. Erro léxico;
3. Erro sintático;
4. Erro semântico

Na Tabela 9, os erros são apresentados e classificados de acordo com o escopo e tipo, juntamente com as informações de descrição para correção. Um código para cada erro foi

proposto no qual o primeiro dígito identifica o tipo de erro, o segundo o escopo em que este se aplica e o terceiro a identificação do erro.

Tabela 9 – Lista de avisos, erros léxicos, sintáticos e semânticos para a linguagem SWRL

Código	Tipo	Escopo	Descrição
111	Aviso	Regra	Antecedente vazio
112	Aviso	Regra	Consequente vazio
113	Aviso	Regra	Antecedentes iguais em regras diferentes
114	Aviso	Regra	Consequentes iguais em regras diferentes
115	Aviso	Regra	Regra duplicada
116	Aviso	Regra	Nome de regra duplicado
121	Aviso	Átomo	Átomo do tipo <i>Built-in</i> no consequente
122	Aviso	Átomo	Átomo duplicado
221	Léxico	Átomo	O predicado não é um termo válido (ocorre quando o átomo não é do tipo <i>Same/Different</i> ou <i>Built-in</i> e o predicado não corresponde a um termo definido na ontologia)
231	Léxico	Argumento	Nome de variável inválido (contendo espaços ou caracteres especiais)
311	Sintático	Regra	Regra não possui o separador entre antecedente e consequente
312	Sintático	Regra	Regra possui disjunções, negações ou símbolos desconhecidos
321	Sintático	Átomo	Número de argumentos inválidos (cada tipo de átomo define um número de argumentos)
331	Sintático	Argumento	Argumento inválido para um determinado átomo (além de definir a quantidade de argumentos, os tipos de átomos definem o tipo de argumento e sua ordem)
332	Sintático	Argumento	Variável usada como indivíduo e tipo de dado simultaneamente
333	Sintático	Argumento	Variável ocorre no consequente e não ocorre no antecedente
411	Semântico	Regra	Regra em conflito com a ontologia (ocorre quando a regra contradiz a ontologia)
412	Semântico	Regra	Dependência circular entre regras
413	Semântico	Regra	Regras contraditórias
421	Semântico	Átomo	Átomos contraditórios na mesma regra (resulta em uma regra que nunca é satisfeita)

Após a identificação dos possíveis tipos de erros em regras SWRL, a próxima etapa consistiu na codificação do mecanismo de detecção de erros. Com isso, a API passou a disponibilizar recursos para identificação e notificação de avisos, erros léxicos e sintáticos. A identificação de erros semânticos não foi implementada, ficando como um dos trabalhos futuros sugeridos.

Como teste comparativo, os vários tipos de erros possíveis foram simulados e executados na SWRL Editor API e também nas ferramentas SWRL Tab, Axiomé e ACE View. Tais resultados são apresentados na Tabela 10. Para cada possibilidade analisada, três opções foram consideradas, sendo elas: (1) não detecta; (2) detecta; (3) não permite. Essa última denota o fato da ferramenta em questão não permitir a ocorrência ou simulação do erro, o que é melhor do que apontar o erro. Nos testes realizados com a API, não existia a possibilidade de não permitir um erro, pois para este teste a interface da ferramenta não foi usada.

Tabela 10 – Comparativo dos recursos de identificação de erros das ferramentas SWRL

Erro	SWRL Editor API	SWRL Tab	Axiomé	ACE View
111	Detecta	Não detecta	Não permite	Detecta
112	Detecta	Não detecta	Não permite	Detecta
113	Detecta	Não detecta	Não detecta	Não detecta
114	Detecta	Não detecta	Não detecta	Não Detecta
115	Detecta	Não detecta	Não detecta	Não Detecta
116	Detecta	Detecta	Detecta	Detecta
121	Detecta	Não detecta	Não detecta	Não permite
122	Detecta	Não detecta	Não detecta	Não permite
221	Detecta	Detecta	Detecta	Detecta
231	Detecta	Detecta	Detecta	Não permite
311	Detecta	Detecta	Não permite	Detecta
312	Detecta	Detecta	Não permite	Detecta
321	Detecta	Detecta	Detecta	Não permite
331	Detecta	Não detecta	Não detecta	Não permite
332	Detecta	Não detecta	Não detecta	Não permite
333	Detecta	Detecta	Detecta	Não permite
411	Não detecta	Não detecta	Não detecta	Não detecta
412	Não detecta	Não detecta	Não detecta	Não detecta
413	Não detecta	Não detecta	Não detecta	Não detecta
421	Não detecta	Não detecta	Não detecta	Detecta

Embora os principais erros sejam tratados, os resultados obtidos demonstram que as ferramentas observadas permitem que os desenvolvedores cometam alguns equívocos durante a composição das regras. Dos erros semânticos, apenas a ferramenta ACE View oferece suporte a um deles (421) ao não permitir a geração de uma regra contraditória.

Por utilizar *templates* no processo de criação e edição das regras, a ferramenta Axiomé, apesar de engessar o processo de composição, não permite cometer alguns erros (111, 112, 311, 312). Do mesmo modo, a ferramenta ACE View não permite a simulação de um grande

número de erros (121, 122, 231, 321, 331, 332 e 333), pois estes se referem a detalhes específicos da sintaxe que são eliminados no emprego da linguagem natural.

4.8 SWRL Editor

A ferramenta SWRL Editor integra as representações visuais, técnicas desenvolvidas e as demais funcionalidades previstas neste trabalho. Ela foi desenvolvida como um *plug-in* da ferramenta Web Protégé. A arquitetura baseada em *plug-ins* é uma das principais funcionalidades do Protégé, pois esse mecanismo de extensão permite que novas funcionalidades sejam adicionadas gradativamente, o que tornou o Protégé uma sólida plataforma para o desenvolvimento de tecnologias semânticas (Tudorache, Vendetti e Noy, 2008). Os *plug-ins* tornam possível utilizar novas bibliotecas e permitem, a adição de novas abas (*tabs*) contendo um ou mais componentes reutilizáveis (*portlets*) na ferramenta Web Protégé.

O SWRL Editor passou a ser desenvolvido após a investigação das ferramentas de regras e a implementação da SWRL Editor API com as técnicas e recursos anteriormente apresentados. Optou-se primeiramente pelo desenvolvimento de protótipos de baixa fidelidade, de modo a facilitar a captação de novos requisitos e permitir a simulação de diversos cenários de uso. Foram utilizadas as tecnologias Java e *Google Web Toolkit* (GWT) na construção do *plug-in* que representa a ferramenta e consiste em uma aba e dois componentes reutilizáveis: (1) *Rule Properties*; e (2) *Rule Panel*, ilustrados na Figura 22.

Figura 22 – Tabs e portlets da ferramenta SWRL Editor

A portlet *Rule Properties* é subdividida em *View*, *Composer*, *Information* e *Options*. As duas primeiras ativam respectivamente os modos de visualização e composição no *Rule Panel* e exibem funções para filtrar as regras, quando no modo *View*, e para compor regras quando no modo *Composer*. O painel *Information* exibe as informações geradas sobre o conjunto de regras e o *Options* deverá conter opções para customização da ferramenta¹⁵.

4.8.1 Visualização

As soluções desenvolvidas para aprimorar a visualização e compreensão das regras SWRL consistem em representações visuais de uma regra, técnicas de organização, recursos para apresentação do conjunto de regras e mecanismo de busca. Na Figura 23 é ilustrada o SWRL Editor no modo de visualização de regras.

¹⁵ Algumas funcionalidades embora previstas, não foram implementadas, como o painel *Options*. Nesses casos, será utilizado o tempo verbal futuro ou mencionada no texto como não desenvolvido.

Figura 23 – Tela da ferramenta SWRL Editor no modo de visualização

O painel *Rule Properties*, na opção *View*, além de ativar o modo de visualização, disponibiliza as seguintes opções de busca e filtro:

- Busca simples por palavra chave;
- Busca avançada, definindo os critérios de busca: nome do objeto de busca (nome da regra ou em um dos tipos de átomos) e em qual parte da regra (antecedente ou consequente) a palavra chave deve ser localizada;
- Filtro por um conceito definido na ontologia;

Essas opções podem ser combinadas e usadas repetidamente com o operador lógico “AND” para formar buscas mais específicas.

O *Rule Panel* no modo de visualização contém três abas: *List*, *Groups* e *Decision Tree*. Na aba *List* são listadas as regras utilizando uma das representações visuais descritas na Seção 3.5.2 e possibilita alternar entre elas; na aba *Groups* as regras são apresentadas de maneira agrupada segundo os critérios descritos na Seção 4.6; e na aba *Decision Tree* é apresentada uma árvore de decisão que representa o conjunto de regras, na qual cada átomo é um nó da árvore (esse recurso está sendo desenvolvido por outro estudante de mestrado do nosso

grupo). O painel exibe se algum filtro está ativo no momento e o número de regras que estão sendo apresentadas, isto é, que obedecem aos critérios determinados no filtro.

Para cada regra apresentada (individualmente) são apresentadas as seguintes opções: 1 - Editar; 2 - Criar uma cópia e editar; e, 3 - Visualizar as regras similares, apresentadas em uma janela *pop-up*.

A ferramenta foi desenvolvida de modo que seja possível empregar mais de um algoritmo para a criação dos grupos e das árvores de decisão. Dessa forma, o usuário da ferramenta pode escolher:

- O algoritmo de agrupamento que deseja utilizar e, se o algoritmo permitir, definir quantos grupos deseja obter;
- O algoritmo para a geração da árvore de decisão que desejará aplicar;

Para tanto, foram definidas interfaces para o agrupamento e árvore de decisão, decorrente da solicitação dos desenvolvedores do BMIR, pois assim, eles podem desenvolver e adicionar novas técnicas à ferramenta de modo dinâmico (*plug-and-play*). As configurações para estas opções poderão ser definidas no painel *Properties* na aba *Options*.

4.8.2 Composição

O processo de criação de regras é ativado na ferramenta quando o usuário seleciona a opção *Composer* ou opta por editar uma regra específica. A *portlet Properties* exibe os termos disponíveis na ontologia que podem ser selecionados pelos usuários. Quando um deles é selecionado, uma janela é aberta com as opções de adicioná-lo como um átomo no antecedente ou consequente. Na Figura 24 é apresentada uma tela da ferramenta SWRL *Editor* no modo composição.

A *portlet Rule Panel*, no modo de composição, disponibiliza três modos de trabalho:

- *Editor* (ativo na Figura 24) – Formulário que utiliza a representação hierárquica para apresentar e organizar os átomos da regra. Divide o ambiente de desenvolvimento nas seguintes janelas de trabalho:
 - Caixa de propriedades (*Properties*), contendo as propriedades do átomo que o desenvolvedor selecionar;
 - Lista de variáveis (*Variables*) utilizadas na regra;
 - Átomos do antecedente (*Antecedent Atoms*), contendo os átomos presentes na parte antecedente da regra. Os átomos são apresentados conforme a representação hierárquica e podem ser alterados na caixa de propriedades;

- Átomos do consequente (*Consequent Atoms*), contendo os átomos presentes na parte consequente da regra. Os átomos são apresentados conforme a representação hierárquica e podem ser alterados na caixa de propriedades;
- *Text* – Editor de regras em linguagem natural, que permitirá compor regras utilizando expressões em inglês. Para tanto, pretende-se utilizar o ACE (Kaljurand, 2008).
- SWRL – Editor SWRL com *Highlight* (descrito na Seção 4.4.1), no qual o desenvolvedor escreve a regra utilizando a sintaxe SWRL. Os átomos e argumentos são apresentados conforme o padrão de cores adotado na visualização SWRL *Highlight*;

Figura 24 – Tela da ferramenta SWRL Editor no modo composição

No topo da *portlet*, é possível definir ou alterar o nome da regra. O botão salvar (*Save*) fica desativado enquanto são identificados erros na regra em desenvolvimento. Além disso, quando o desenvolvedor opta por voltar ao modo de visualização sem salvar a regra, a ferramenta exibe uma mensagem de confirmação.

Na parte inferior, são apresentadas as seguintes abas:

- Problemas (*Problems*) – contem as possíveis notificações de erros e avisos para o desenvolvedor da regra, descritos na Seção 4.7;
- Regras similares (*Similar Rules*) – Lista de regras similares à regra em desenvolvimento, conforme o algoritmo descrito na Seção 4.5, de modo que o desenvolvedor pode reutilizar os átomos destas regras;

- Termos sugeridos (*Suggest Terms*) – conterá uma lista com os termos sugeridos.

Essas abas funcionam de maneira independente do modo de composição utilizado, porém as regras similares e os termos sugeridos são apresentados na representação correspondente ao modo utilizado.

4.8.3 Comparativo entre Ferramentas

Diferentemente das ferramentas SWRL Tab e Axiomé, que possuem seus recursos acoplados ao Protégé 3.x, a ferramenta SWRL *Editor* utiliza os recursos disponíveis na SWRL Editor API, que foram desenvolvidos para ser independente de plataforma (Protégé 3.x ou 4.x).

Ao mesmo tempo em que a ferramenta abstrai os detalhes da linguagem SWRL com a visualização hierárquica, não limita os desenvolvedores avançados, pois disponibiliza o acesso direto a sintaxe da regra por meio da visualização com *highlight*.

A ferramenta ACE View está disponível apenas para a versão 4.x do Protégé e permite a criação de regras SWRL em linguagem natural. Para utilização da ferramenta é necessário conhecer a linguagem ACE. Pretende-se no futuro integrar as funcionalidades da ferramenta ACE View à ferramenta SWRL Editor.

Um comparativo dos principais recursos disponíveis nas ferramentas SWRL Editor, SWRL Tab, Axiomé e ACE View são apresentados na Tabela 11. O fato de estar disponível na Web, e de integrar diversos recursos (e suas melhorias) em uma única ferramenta faz com que a ferramenta SWRL *Editor* seja uma importante contribuição deste trabalho.

Tabela 11 – Comparativo das interfaces e funcionalidade disponíveis nas ferramentas de regras SWRL

Interface/Recurso	SWRL Editor	SWRL Tab	Axiomé	ACE View
Plataforma	Web	Desktop	Desktop	Desktop
Tabela de decisão	-	-	-	-
Árvore de decisão	Previsto	-	-	-
Diagrama gráfico	-	-	Possui	-
Editor de texto	Possui	Possui	Possui	-
Formulário/ <i>Template</i>	Possui	-	Possui	-
Linguagem natural	Previsto	-	Parcialmente	Possui
Representação visual	Possui	-	Possui	-
Grupos/Categorias	Possui	-	Possui	-
Sugestão de termos	Previsto	Possui	-	-
Mecanismo de busca	Possui	Possui	-	Possui
Informações gerais	Possui	-	-	-

Interface/Recurso	SWRL Editor	SWRL Tab	Axiomé	ACE View
sobre as regras				
Customização	Previsto	-	-	-

4.9 Considerações Finais

Nesse capítulo foi apresentado um conjunto de recursos cujo objetivo é aprimorar a manipulação e a compreensão de regras SWRL. O desenvolvimento da SWRL Editor API juntamente com as representações visuais e as técnicas de análise de similaridade, agrupamento e detecção de erros contribuem para a melhoria do processo de exploração e composição de regras SWRL.

Além disso, é apresentada uma ferramenta baseada na Web, denominada *SWRL Editor*, desenvolvida como um *plug-in* da ferramenta Web Protégé, que incorpora estes recursos para auxiliar na criação e na manipulação de regras SWRL. Embora a ferramenta disponibilize novos recursos, estes ainda são voltados para desenvolvedores que conhecem a linguagem SWRL.

5 Estudo de Caso: *Autism Phenologue Rules*

Neste capítulo, são descritos os resultados obtidos durante o desenvolvimento do estudo de caso usando as regras da ontologia *Autism Phenologue Rules*, versão 3.4.4 de março de 2010. A partir desse estudo, foi possível propor uma nova representação visual que atende especialmente as regras dessa ontologia e abstrai os detalhes da linguagem SWRL, priorizando os conceitos utilizados nas anotações.

A escolha da *Autism Phenologue Rules* para este estudo de caso é motivada pelos seguintes fatores:

1. A ontologia possui um grande número de regras SWRL;
2. As regras são grandes, ou seja, possuem diversos átomos tanto no antecedente quanto no consequente;
3. Temos contato direto com os desenvolvedores das regras e pesquisadores do autismo (através do BMIR);

5.1 Introdução

O autismo é uma série de transtornos complexos no desenvolvimento mental, caracterizados por deficiências sociais, dificuldades de comunicação e padrões restritos, repetitivos e estereotipados de comportamento (Frith e Hill, 2003). Sua escolha para o projeto Phenologue se deve ao fato da não identificação dos genes responsáveis pela doença, além de estudos recentes demonstrarem o crescimento da ocorrência em casos de autismo na população (Tu et al., 2008).

O projeto Phenologue, coordenado pelo professor Amar Das (*Stanford University*), tem por objetivo buscar colaborativamente adquirir, gerenciar, e inferir novas informações a cerca de fenótipos do autismo a partir de dados experimentais e resultados publicados na literatura (Hassanpour, O'Connor e Das, 2010). O projeto envolve especialistas em autismo e informática da universidade e do *National Database for Autism Research* (NDAR), que é também um repositório de dados de pesquisas desenvolvido pelo *National Institutes of Health* (NIH) para apoiar e acelerar o avanço da pesquisa sobre *Autism Spectrum Disorders* (ASD) (NDAR, 2010). Dentro deste projeto, está sendo elaborada e criada a ontologia *Autism*

Phenologue Rules. Esta ontologia estende ontologias disponíveis na *OBO Foundry* (<http://www.obofoundry.org/>) e sua criação ocorreu por meio de uma revisão sistemática da literatura para identificar termos do domínio e relacionamentos relevantes para fenótipos de autismo (Tu et al. 2008).

Para realizar a categorização dos fenótipos de autismo foi utilizada a linguagem SWRL, de modo que, a cada caracterização os pesquisadores criam regras para descrevê-lo. A colaboração com os envolvidos no projeto *Phenologue* contribuiu na definição dos requisitos e análise dos resultados obtidos. Eles também tiveram acesso a ferramenta para utilização no desenvolvimento de novas regras e manutenção das regras atuais.

5.2 Conjunto de Regras

Foi feita uma análise das principais características do conjunto de regras, utilizado a SWRL Editor API, com o objetivo de conhecer em detalhes as regras. Na Tabela 12 são apresentadas as informações pertinentes as 156 regras presentes na ontologia.

Tabela 12 – Visão geral das regras disponíveis na ontologia *Autism Phenologue Rules*

Informação	Total	Antecedente	Consequente
Número de regras	156	-	-
Número de átomos	2137	994	1143
Número de predicados distintos	155	147	9
Média de átomos por número de regras	13	6	7
Número mínimo e máximo de átomos por regra	[10, 20]	[5, 15]	[5, 8]
Número de argumentos	3890	1877	2009
Número de argumentos distintos	1304	997	1013
Média de argumentos por número de regras	24	12	12
Média de argumentos distintos por número de regras	8	6	6
Número mínimo e máximo de argumentos por regra	[18, 46]	[9, 37]	[9, 17]
Número mínimo e máximo de argumentos distintos por regra	[9, 18]	[5, 16]	[5, 10]

O grande número de átomos presente nas regras contrasta com a quantidade de predicados utilizados. Isso revela que embora a média de átomos por regra seja grande (13 no total, 6 no antecedente e 7 no consequente), se comparado com outros conjuntos, os predicados se repetem com certa frequência nas regras. Tal observação é acentuada quando observado apenas os consequentes das regras, pois todas as regras utilizam apenas 9 predicados distintos como consequentes. A menor regra possui 10 átomos (distribuídos igualmente entre

antecedente e consequente) e a maior regra 20 átomos, com o consequente tendo no máximo 8 átomos.

Para enriquecer a análise dos átomos e predicados, na Figura 25 é ilustrada a distribuição do número de átomos no antecedente, consequente e na regra. A distribuição revela que a moda do número de átomos no antecedente é idêntica à média, assim, são poucas as regras que não possuem 6 átomos no antecedente. A análise da distribuição dos átomos no consequente indica uma grande quantidade de regras com 5 (26 regras), 7 (28 regras) e 8 átomos (90 regras) nesta parte da regra. O gráfico contendo todos os átomos da regra segue a curva de distribuição do consequente, uma vez que os antecedentes possuem poucas variações.

Figura 25 – Distribuição dos átomos nas regras, antecedentes e consequentes

A lista dos principais predicados (presentes em mais de 10 regras) e a quantidade de ocorrências no antecedente é apresentada na Figura 26. O único predicado no antecedente, que ocorre em todas as regras é o `swrlx:createOWLThing`, responsável por criar um indivíduo na regra vinculando-o a outros indivíduos, permitindo assim, condições existenciais

nas regras SWRL. Na Figura 27 são apresentados todos os predicados que ocorrem no consequente e o número de regras em que são utilizados. Desses, 3 predicados aparecem em todas as regras (*is derived from*, *temporal:hasValidTime* e *Phenotype record*) e, dos demais predicados do consequente, apenas o *adi-r2003:SubjectKey* aparece em poucas regras. Porém, utilizando o mecanismo de busca disponível na ferramenta foi possível identificar que apenas este predicado ocorre em ambas as partes da regra (antecedente e consequente).

Figura 26 – Quantidade de ocorrência dos principais predicados que ocorrem no antecedente das regras da ontologia *Autism Phenologue Rules*

Figura 27 – Quantidade de todos os predicados que ocorrem no consequente das regras da ontologia *Autism Phenologue Rules*

A distribuição dos átomos/predicados sob a perspectiva de seus tipos é apresentado na Tabela 13. Das 156 regras, nenhuma possui átomos do tipo *Data range* e *Same/different*, os átomos do tipo *Object property* não ocorrem no antecedente e os *Built-ins* no consequente, sendo que apenas átomos do tipo *Class* e *Data valued property* ocorrem em ambas as partes. Além disso, é observado que toda regra possui pelo menos 2 átomos do tipo *Class*, 1 dos tipos *Object property* e *Built-in*, e 5 do tipo *Data valued property*.

Tabela 13 – Distribuição dos tipos de átomos das regras do autismo

Tipo de átomo	Antecedente	Consequente	Regra			
			Total	Média	Máx.	Mín.
<i>Class</i>	156	255	411	2.6	3	2
<i>Object property</i>	0	303	303	1.9	2	1
<i>Data valued property</i>	510	542	1052	6.7	12	5
<i>Data range</i>	0	0	0	0	0	0
<i>Same/different</i>	0	0	0	0	0	0
<i>Built-in</i>	328	0	328	2.1	4	1

Com relação aos argumentos, as informações apresentadas na Tabela 12 indicam que é pequena a quantidade argumentos distintos por regra se comparado ao número total, contudo, com apenas estes dados não é possível extrair informações relevantes, sendo necessário analisar em paralelo a distribuição dos tipos de argumento, que é apresentada na Tabela 14.

Tabela 14 – Distribuição dos tipos de argumentos das regras do autismo

Tipo de argumento	Antecedente	Consequente	Regra			
			Total	Média	Máx.	Mín.
<i>Individuals</i>	0	147	147	0.9	1	0
<i>Data Literals</i>	41	156	197	1.3	3	1
<i>Individual variables</i>	666	1256	1922	12.3	17	9
<i>Data variables</i>	1170	386	1556	10	25	6
Variáveis	1836	1642	3478	22.3	-	-
d-Object	1211	542	1753	11.2	-	-
i-Object	666	1403	2069	13.2	-	-

Os principais tipos de argumentos encontrados nas regras são variáveis, referentes em primeiro lugar aos indivíduos e em segundo aos tipos de dados. Referências a indivíduos presentes na ontologia ocorrem apenas no consequente e os valores literais são utilizados com mais frequência no consequente.

O estudo e análise dessas informações proporcionou uma visão geral e completa do conjunto de regras da ontologia *Autism Phenologue Rules*. Tal fato contribuiu para a criação

de uma representação visual específica para essas regras. Além disso, esse tipo de análise permite comparar conjuntos de regras sob as perspectivas observadas e viabiliza a categorização de conjuntos de regras a partir destes dados. Durante o período do projeto outros conjuntos de regras SWRL foram analisados, mas, nenhum deles apresentou regras complexas e com características semelhantes às encontradas na ontologia *Autism Phenologue Rules*. Como exemplo, o conjunto de regras *Family History* (Peace e Brennan, 2008) possui 146 regras, nas quais o número total de átomos é 470 (média de 3 átomos por regra) e o número distinto de predicados é 181 (mais do que o número encontrados nas regras do autismo, cujo total de átomos totaliza 2137).

5.3 Representação Visual das regras do autismo

As representações visuais desenvolvidas neste trabalho têm por objetivo facilitar e ampliar o entendimento das regras. Enquanto a visualização SWRL *Highlight* melhora a disposição dos átomos e enriquece a visualização com cores, a visualização hierárquica oculta alguns detalhes da sintaxe e, além de cores, agrega símbolos para cada tipo de átomo.

Pelo fato de as regras do autismo apresentarem uma estrutura homogênea e uma quantidade significativa de átomos, o ganho é ainda maior. Visualmente, as regras são mais bem apresentadas, o que permite ao observador identificar, de maneira objetiva, os padrões que a regra possui, principalmente quando a visualização hierárquica é usada.

Na Figura 28 é apresentada a visualização com *Highlight* (Seção 4.4.1) da regra **ConcludeSavantNegative_15756693** (uma das maiores regras do conjunto) e na Figura 29 a visualização hierárquica (Seção 4.4.2) da mesma regra. A primeira distingue os tipos de átomos utilizados, as variáveis e os valores literais. A segunda, além de organizar os átomos em uma estrutura visualmente significativa, oculta alguns detalhes da sintaxe, como é o caso de alguns átomos do tipo *built-ins* que foram apresentados com os respectivos operadores matemáticos. Para efeito de comparação, na Figura 30 é apresentada a visualização hierárquica da regra **ConcludeDelayedPhrases_1185572**, na qual é possível verificar a correspondência dos átomos consequentes, alterando apenas dois valores de argumentos literais.

▼ **ConcludeSavantNegative_15756693**

```

ADI-2003(?a) ^
adi-r2003:ADI_2003_genbehav_cvisspz(?a, ?ssf1) ^
adi-r2003:ADI_2003_genbehav_evisspz(?a, ?ssf5) ^
adi-r2003:ADI_2003_genbehav_cmemz(?a, ?ssf2) ^
adi-r2003:ADI_2003_genbehav_ememz(?a, ?ssf6) ^
adi-r2003:ADI_2003_genbehav_cmusicz(?a, ?ssf3) ^
adi-r2003:ADI_2003_genbehav_emusicz(?a, ?ssf7) ^
adi-r2003:ADI_2003_genbehav_ccompuz(?a, ?ssf4) ^
adi-r2003:ADI_2003_genbehav_ecompuz(?a, ?ssf8) ^
swrlb:add(?sum, ?ssf1, ?ssf2, ?ssf3, ?ssf4, ?ssf5, ?ssf6, ?ssf7, ?ssf8) ^
swrlb:divide(?score, ?sum, 8) ^
swrlb:lessThan(?score, 0.16) ^
adi-r2003:SubjectKey(?a, ?subjectID) ^
adi-r2003:ADI_2003_interview_date(?a, ?date) ^
swrlx:createOWLThing(?phenorecord, ?subjectID)
->

Phenotype record(?phenorecord) ^
temporal:hasValidTime(?phenorecord, ?date) ^
is derived from(?phenorecord, "ConcludeSavantNegative") ^
autism-core:subjectId(?phenorecord, ?subjectID) ^
autism-core:subject_has_quality_or_disposition(?phenorecord, Savant negative)

```

Figura 28 – Representação visual SWRL *highlight* da regra ConcludeSavantNegative_15756693

Figura 29 – Representação visual hierárquica da regra ConcludeSavantNegative_15756693

Figura 30 – Representação visual hierárquica da regra **ConcludeDelayedPhrases_118572**

5.3.1 Representação visual do autismo

Durante a exploração das regras nas visualizações SWRL *highlight* e hierárquica foram identificados padrões de estruturas nas regras que possibilitaram desenvolver uma representação específica para este conjunto, denominada de visualização do autismo. Inicialmente, foi possível identificar que todas as regras compartilhavam determinados átomos, sendo que apenas uma pequena parte (alguns predicados ou argumentos) é que distinguem cada uma das regras. Tomando como exemplo a regra **ConcludeDelayedPhrases_118572**, na Figura 31 é ressaltada apenas a parte distinta dessa regra, nesse caso, o antecedente possui dois átomos distintos e o consequente apenas dois argumentos.

Figura 31 – Partes distintas da regra ConcludeDelayedPhrases_1185572

Quanto à parte comum das regras, foram identificadas poucas variações de estrutura (Apêndice C). Na Figura 32 e Figura 33 são apresentadas duas dessas estruturas comuns denominadas de A e B respectivamente. Nesse caso, a estrutura A é complementar à estrutura da Figura 31, de modo que juntos formam a regra **ConcludeDelayedPhrases_1185572**.

Figura 32 – Estrutura comum A para as regras do autismo

Figura 33 – Estrutura comum B para as regras do autismo

Após a identificação dessas estruturas, foi percebido que alguns átomos do antecedente estão ligados a átomos do consequente por meio de variáveis, ou seja, se referem ao mesmo argumento. Tal padrão foi observado em todas as estruturas, para ilustrar tal aplicação é apresentado, na Tabela 15, o mapeamento realizado entre os termos do antecedente e do consequente encontrados nas estruturas comuns A e B.

Tabela 15 – Mapeamento dos termos do antecedente com consequente para as estruturas comuns A e B

Estrutura comum A		
Variável	Predicado Antecedente	Predicado Consequente
?subjectID	adi-r2003:SubjectKey swrlx:createOWLThing	autism-core:subjectId
?date	adi-r2003:ADI_2003_interview_date	temporal:hasValidTime
?phenorecord	swrlx:createOWLThing	Phenotype Record
Estrutura comum B		
Variável	Predicado Antecedente	Predicado Consequente
?subjectID	adi-r2003:SubjectKey swrlx:createOWLThing	autism-core:subjectId
?date	adi-r2003:ADI_2003_interview_date	temporal:hasValidTime
?phenorecord	swrlx:createOWLThing	Phenotype Record
?qv	swrlx:createOWLThing	has measurement Quantitative Value has value

Dessa forma, após identificar as estruturas comuns e um padrão entre essas estruturas e partes específicas das regras, foi elaborada uma representação visual capaz de representar adequadamente e objetivamente as estruturas e padrões encontrados para este conjunto de regras. Na Figura 34 é apresentada a representação visual gerada para a regra **ConcludeDelayedPhrases_1185572** (correspondente à estrutura comum A), nesse caso a parte específica do antecedente da regra é apresentada no campo *match* e do consequente nos valores dos campos *subject has quality or disposition* e *is derived from*.

Figura 34 – Representação visual do autismo da regra ConcludeDelayedPhrases_1185572 correspondente a estrutura comum A

As cores e símbolos utilizados buscam distinguir os tipos dos átomos, o local onde eles se encontram na regra, e a que conceito se referem. Os nomes das variáveis foram omitidos (?a, ?d1, etc), o átomo do tipo *Class* do antecedente é destacado no canto superior esquerdo da representação. Na parte superior (da linha) só aparecem as condições de casamento (*matching*) que separam essa regra das demais. Na parte inferior, aparecem todos os indivíduos criados quando a regra dispara (na Figura 34 apenas um indivíduo é criado) e suas propriedades. Cada propriedade deve ser preenchida com o mesmo valor da propriedade indicada do átomo com o qual foi feito o *match* (no nosso exemplo, um átomo do tipo *Class* ADI-2003) ou por constantes. Para que uma regra dispare, é necessário que o *match* ocorra e que todas as propriedades a serem preenchidas ou existam no átomo que fez o *match* ou sejam constantes.

Na Figura 35, a regra **MeasureSocialPlay_11857572** (correspondente à estrutura comum B) é apresentada na representação do autismo. Nesse padrão, é acrescentado um segundo losango referente ao átomo do tipo *Class* denominado de *Quantitative Value* e o valor do *match* é utilizado como parâmetro para o predicado *has value* do consequente, que faz referência ao mesmo argumento.

Figura 35 – Representação visual do autismo da regra MeasureSocialPlay_11857572 correspondente a estrutura comum B

Os desenvolvedores da regra do autismo validaram a representação desenvolvida e relataram que a mesma é adequada para representar a regra sem perda de significado. Com essa representação, os detalhes da sintaxe SWRL foram omitidos e, com isso, as regras se tornaram mais simples e mais adequadas para serem apresentadas para pesquisadores biomédicos não técnicos em computação. Por fim, todas as regras foram analisadas manualmente e não foram identificados distorções ou problemas nessa representação. Ao permitir que os desenvolvedores das regras do autismo, que na sua grande maioria são médicos, possam visualizá-las em um nível de abstração própria, esta representação é uma das principais contribuições deste trabalho.

5.4 Similaridade entre as regras

O fator de similaridade entre duas regras, gerado pela SWRL Editor API, é utilizado na ferramenta em quatro momentos distintos:

1. **Durante a visualização das regras**, quando é possível visualizar as regras similares a uma determinada regra;
2. **Na formação dos grupos**, quando esse fator é utilizado pelo algoritmo de agrupamento;
3. **Durante a composição das regras**, quando é possível visualizar as regras similares e reutilizar os átomos dessas regras;
4. **Durante a detecção de erros**, quando se utiliza a similaridade para detectar regras ou partes da regra repetidas;

Com o objetivo de ter uma noção geral das distâncias entre as regras, foi feito o cálculo da distância entre todos os pares de regras (12090 pares, para um total de 156 regras). Para isso, foram consideradas seis políticas para determinar o vetor de características das regras.

São elas:

1. Todos os predicados da regra (Política 1);
2. Somente os predicados do antecedente (Política 2);
3. Somente os predicados do consequente (Política 3);
4. Todos os átomos da regra (Política 4);
5. Somente os átomos do antecedente (Política 5);
6. Somente os átomos do consequente (Política 6);

Os resultados obtidos são apresentados na Tabela 16 (para as políticas 1, 2 e 3) e na Tabela 17 (para as políticas 4, 5 e 6). Foram observados o menor e o maior grau de similaridade, além da moda e do número de graus distintos que ocorreram em cada situação. Os valores entre parênteses correspondem à quantidade de vezes que o grau ocorreu.

Enquanto a análise de similaridade utilizando o predicado não considera os valores dos argumentos, na análise utilizando os átomos tais valores são considerados, resultando em um intervalo maior de graus obtidos (4 para a política 3, enquanto 6 para a política 6). Além disso, no primeiro caso a obtenção do valor 0 indica que as regras ou suas partes utilizam exatamente os mesmos conceitos, já o segundo caso indica que as regras ou suas partes são idênticas. Dessa forma, como já constatado, a maior parte das regras possui os mesmos

conceitos no consequente (moda 0 na política 3). O grau que ocorreu com mais frequência (1,41 – Políticas 1, 2 e 5) é baixo, mostrando que o conjunto de regras como um todo é bastante homogêneo. Quando observada a igualdade das regras, foram encontradas 2 regras repetidas (Política 4), 5 pares de regras com os mesmos antecedentes (Política 5) e 10 pares com o mesmo consequente (Política 6). A identificação desses problemas é uma importante contribuição deste trabalho e são abordados com mais detalhes na Seção 5.6, uma vez que também foram identificados durante a detecção de erros.

Tabela 16 – Resumo dos cálculos de similaridade entre as regras do conjunto *Autism Phenologue Rules*, considerando os predicados das regras

	Predicado		
	Regra (Política 1)	Antecedente (Política 2)	Consequente (Política 3)
Mínimo:	0 (17)	0 (17)	0 (7621)
Moda:	1,41 (4570)	1,41 (4588)	0 (7621)
Máximo:	5,0 (27)	4,35 (56)	2,23 (1089)
Quantidade de valores distintos para a distância:	24	20	4

Tabela 17 - Resumo dos cálculos de similaridade entre as regras do conjunto *Autism Phenologue Rules*, considerando os átomos das regras

	Átomo		
	Regra (Política 4)	Antecedente (Política 5)	Consequente (Política 6)
Mínimo:	0 (2)	0 (5)	0 (10)
Moda:	2,44 (4176)	1,41 (4547)	2,0 (7141)
Máximo:	5,09 (54)	4,47 (2)	3,0 (1089)
Quantidade de valores distintos para a distância:	24	20	6

As regras com distância menor que 2 são consideradas muito semelhantes. Contudo, as regras do autismo podem ser consideradas similares entre si, se a distância for até 3, dado o grande número de átomos nas regras. Tal constatação foi realizada por meio da análise subjetiva das combinações geradas e não podem ser generalizada para todos os conjuntos de regras.

5.5 Agrupamento

A técnica de agrupamento desenvolvida neste trabalho foi empregada com o objetivo de agrupar as regras de acordo com a similaridade ao invés da estrutura de regras (os tipos de

átomos que a regra possui), utilizado pela ferramenta Axiomé. O algoritmo escolhido para realizar o agrupamento foi o K-means (Jain, Murty e Flynn, 1999), pela sua simplicidade (a facilidade de implementação). Como desvantagens, pode-se citar: o fato de ele não ser determinístico¹⁶; a necessidade de se definir o número de agrupamentos a serem obtidos; e a necessidade de se executar o algoritmo novamente no caso de modificação ou adição de um novo elemento (no nosso caso, uma regra) exige o reagrupamento.

Para avaliar a abordagem apresentada neste trabalho, foi feito um estudo comparativo entre os agrupamentos gerados com a ferramenta Axiomé e com a abordagem apresentada para o conjunto de regras *Autism Phenologue Rules*.

Na Tabela 18 são apresentados os grupos gerados pela ferramenta Axiomé e a quantidade de regras que cada grupo contém. Tais características não são controladas pelo desenvolvedor, o que pode levar a grupos com apenas uma regra (4 grupos) ou grupos muito numerosos (um grupo com 121 regras), pois é a estrutura da regra que determina a qual grupo ela pertence. Durante o agrupamento é gerada uma expressão canônica com números e símbolos que identificam a estrutura das regras e nomeiam o grupo, todavia, as expressões não são intuitivas e dificultam seu uso e memorização.

Tabela 18 – Agrupamento realizado pela ferramenta Axiomé nas regras do autismo

Nome do grupo	Quantidade de regras
(144444)^{(55)}-(@12244)	2
(1444)-(@1224#(14)44)	121
(1444444)^{(55)^{(5)}}-(@12244)	2
(144444)^{(555)}-(@12244)	1
(144444444)^{(55)}-(@12244)	2
(14#(5)4#(5)44)^{(5)}-(12244)	1
(14#(5)44)^{(5)}-(12244)	8
(14#(5)44)-(@12244)	15
(144444444)^{(555)}-(@12244)	1
(1444)-(@12244)	1
(14444444444)^{(55)^{(5)}}-(@12244)	2

A exploração das regras, em seus respectivos grupos, mostra que o uso da estrutura da regra para realizar o agrupamento, agrega pouco valor para o entendimento das mesmas, pois o fator determinante para o agrupamento são os tipos dos átomos utilizados ao invés dos

¹⁶ O não determinismo faz com que o algoritmo possa gerar resultados distintos para o mesmo conjunto de regras.

próprios átomos. No caso específico do autismo, o fato das regras serem muito parecidas entre si (possuírem muitos predicados em comum) contribuiu para que os grupos com poucas regras possuam regras similares. Por exemplo, duas regras idênticas são classificadas no mesmo grupo (pois possuem a mesma estrutura), entretanto se é acrescentado um novo átomo a uma dessas regras, essa passa a pertencer a um novo grupo.

Diferentemente da abordagem utilizada pela ferramenta Axiomé, ao considerar o grau de similaridade na realização dos agrupamentos, o esperado é que os grupos contenham regras similares independentemente da estrutura, logo, utilizando o cenário do exemplo anterior o método desenvolvido classificará as duas regras no mesmo grupo mesmo após a modificação.

Na Tabela 19, é apresentado o conjunto de agrupamentos obtido pela aplicação do algoritmo K-means. Foram realizados dois testes utilizando matrizes de características distintas (predicado e regra) para o cálculo de similaridade. Os grupos receberam nomes sequenciais a partir do número de regras em cada um deles.

Tabela 19 – Agrupamento por similaridade gerado pela aplicação do algoritmo K-means nas regras do autismo

Nome do grupo	Similaridade do Predicado	Similaridade do Átomo
Grupo 1	2	2
Grupo 2	2	2
Grupo 3	2	2
Grupo 4	3	3
Grupo 5	3	4
Grupo 6	3	4
Grupo 7	6	7
Grupo 8	9	9
Grupo 9	15	15
Grupo 10	21	20
Grupo 11	90	88

Quanto mais precisa for a identificação da similaridade, mais apropriada se torna a técnica de agrupamento. Os resultados obtidos com a aplicação do algoritmo K-means geraram, nesse caso, grupos de regras que se referiam aos mesmos conceitos, ressaltando a utilidade do agrupamento por similaridade.

Comparando o uso de similaridade entre predicados e entre átomos, eles resultaram, para esse conjunto de regras, em um número equivalente de valores, pois houve apenas pequenas diferenças na distribuição dos grupos. Porém, a distribuição do número de regras em cada grupo se mostrou mais adequada do que a da ferramenta Axiomé, pelo fato de não possuir grupos com apenas uma regra e o maior grupo possuir um número menor de regras. Além

disso, ao verificar e analisar as regras classificadas por similaridade, os grupos obtidos possuem regras que se referem aos mesmos conceitos ao invés de, necessariamente, compartilhar a mesma estrutura. A aplicação do algoritmo K-means no conjunto de regras *History Family* se mostrou mais satisfatória ainda, uma vez que distribuiu melhor as regras em relação as regras do autismo. Isso se deve ao fato das regras do autismo possuírem muitos átomos/predicados em comum.

Embora o número de grupos utilizados para este teste tenha sido 11, o desenvolvedor das regras pode ajustar o número de grupos que lhe é conveniente. Em consulta aos desenvolvedores do autismo (BMIR) sobre a melhor forma de realizar o agrupamento, foi sugerido que os grupos sejam gerados automaticamente e, a partir desse agrupamento prévio, o usuário possa nomeá-los e modificá-los conforme seu interesse.

5.6 Detecção de Erros

A detecção de erros é realizada após cada modificação em uma regra, durante a realização da tarefa de composição de regras. Na ocorrência de algum problema (Tabela 9), um aviso é apresentado ao desenvolvedor para que ele possa corrigi-lo, caso contrário, uma mensagem de sucesso é apresentada no painel de apresentação dos problemas, sem interferir na interação do usuário.

Como parte do estudo de caso, o algoritmo de detecção de erros foi aplicado para todas as regras da *Autism Phenologue Rules* com a finalidade de identificar possíveis regras com erros. Após a aplicação desse teste, foram encontrados os seguintes problemas:

1. 5 pares de regras com o antecedente idêntico (erro 113);
2. 10 pares de regras com o consequente idêntico (erro 114);
3. Duas regras repetidas (erro 115);
4. Todas as regras acusaram o erro 331, no qual as variáveis **?date** e **?qv** aparecem como variáveis de dois tipos distintos na mesma regra;

Embora os problemas 113, 114 e 115 não sejam propriamente erros, mas avisos, e possivelmente não acarretam problemas na interpretação do conjunto de regras, tais situações podem não ser desejáveis, e, nesse caso, poderiam ter sido evitadas.

O fato de algumas regras possuírem o mesmo antecedente ou o mesmo consequente pode ser proposital, pois a primeira possibilidade pode simplificar o consequente¹⁷ e a segunda supre a falta do operador lógico “OR” no antecedente, que não é suportado pela sintaxe da linguagem SWRL¹⁸. Entretanto, esses avisos podem alertar o desenvolvedor e evitar possíveis equívocos por parte dele. Os pares de regras que possuem antecedentes idênticos são (problema 113):

- **ConcludeNoPhrase_17276746 e ConcludeNoPhrases_11857572;**
- **ConcludePositiveHistoryofRegression2_16598435 e
ConcludePositiveHistoryofRegression_15940295;**
- **MeasureAgeFirstWalkedUnaided_12819447 e
MeasureAgeofAcquisitionofBowelControl_12819447;**
- **MeasureAgeofAcquisitionofPhrases_12819447_15780135 e
MeasureAgeofFirstPhrases_12819447_15756693;**
- **MeasureCurrentDifficultieswithMinorChanges_8178836 e
MeasureCurrentMinorChangesinEnvironment_8178836;**

Os grupos de regras que possuem consequentes idênticos são (problema 114):

- **ConcludeDelayedPhrases_11857572 e ConcludeDelayedPhrases_17276746;**
- **ConcludeNegativeHistoryofRegression_15940295 e
ConcludeNegativeHistoryofRegression_16598435;**
- **ConcludeNoPhrase_17276746 e ConcludeNoPhrases_11857572;**
- **ConcludeNotDelayedPhrases_11857572 e
ConcludeNotDelayedPhrases_17276746;**
- **ConcludePositiveHistoryofRegression1_16598435,
ConcludePositiveHistoryofRegression2_16598435 e
ConcludePositiveHistoryofRegression_15940295¹⁹;**
- **ConcludeQuestionableHistoryofRegression1_15940295 e
ConcludeQuestionableHistoryofRegression2_15940295;**
- **ConcludeSavantNegative_15756693 e ConcludeSavantNegative_17276746;**
- **ConcludeSavantPositive_15756693 e ConcludeSavantPositive_17276746;**

¹⁷ Suponha que as expressões $A \rightarrow B$ e $A \rightarrow C$ são válidas, então a expressão $A \rightarrow B \wedge C$ é equivalente.

¹⁸ Suponha que as expressões $A \rightarrow C$ e $B \rightarrow C$ são válidas, então a expressão $A \vee B \rightarrow C$ é equivalente.

¹⁹ As três regras possuem o mesmo consequente e correspondem a três pares

Tais regras e combinações foram analisadas manualmente, possibilitando realizar as seguintes constatações:

- As regras duplicadas apareceram no antecedente e consequente;
- Os demais pares de regras com antecedente idêntico apresentaram pequenas diferenças no consequente, o que pareceu ter ocorrido intencionalmente. Em um dos pares o diferencial foram os valores dos predicados 'is derived from' e `autism-core:subject_has_quality_or_disposition`, e nos demais apenas o predicado 'is derived from'.
- Com consequente idêntico, as regras com o prefixo **ConcludeDelayedPhrases** apresentaram a seguinte diferença: `?phraseage > 36` e `?phraseage > 33`, com isso é possível afirmar que a segunda engloba a primeira, tornando-a desnecessária. O mesmo ocorreu com as regras cujo prefixo é **ConcludeNotDelayedPhrases** porém, nesse caso, a diferença foi: `?phraseage <= 36` e `?phraseage <= 33`, e, assim, pode-se dizer que a primeira torna a segunda desnecessária. Todavia, as regras **ConcludeDelayedPhrases** e **ConcludeNotDelayedPhrases** são complementares (geram resultados disjuntos) e não poderiam sofrer intersecção como está ocorrendo. Caso o valor da variável `?phraseage` seja igual a 35 e todas as demais condições do antecedente sejam satisfeitas a regra classificaria o indivíduo em ambos os casos.
- As regras com o prefixo **ConcludeSavantNegative** e **ConcludeSavantPositive** apresentaram o mesmo cenário descrito anteriormente.
- Os demais pares de regras com consequente idêntico apresentaram átomos adicionais no antecedente e/ou átomos repetidos com valores diferentes. Neste caso, não é possível afirmar se essas ocorrências foram desejáveis ou não.

Generalizando, o problema de regras duplicadas (problema 115) é mais crítico que os outros dois, uma vez que a redundância, nesse contexto, não oferece nenhuma vantagem para o desenvolvedor. O motivo da ocorrência desse erro, segundo os responsáveis (do BMIR), se dá pelo fato das regras serem desenvolvidas colaborativamente. A hipótese da duplicação das regras por um erro²⁰ de um usuário foi descartada porque as regras possuem nomes que atendem o padrão estabelecido. As regras duplicadas identificadas são:

²⁰ As ferramentas SWRL Tab e Axiomé possuem a opção de duplicar uma regra.

- **ConcludeNoPhrase_17276746** e **ConcludeNoPhrases_11857572**;
- **ConcludePositiveHistoryofRegression2_16598435** e **ConcludePositiveHistoryofRegression_15940295**;

Tais identificações também foram realizadas durante os testes de análise de similaridade, pois, algumas das verificações do algoritmo de detecção de erros utilizam o resultado do método de similaridade. Ao utilizar a ferramenta SWRL Editor esses problemas poderiam ser evitados uma vez que o usuário é alertado quando intencionalmente ou casualmente os gera. Diferente dos demais tipos de erros os avisos não desabilitam o botão salvar e, consequentemente, não impedem que as regras possam ser salvas.

O erro 331, em que a mesma variável aparece relacionada a dois tipos distintos na mesma regra (I-Object e D-Object) eram desconhecidos pelos desenvolvedores biomédicos, que até o momento, estão usando as regras exclusivamente para modelar o domínio de conhecimento. A ferramenta usada por eles, a SWRL Tab do Protégé, não identificou o erro, todavia, em tempo de execução a máquina de inferência JESS (usada pelo Protégé para executar regras SWRL) não executaria as regras como esperado, o que alertaria sobre a existência de algum problema. Mesmo assim, a detecção durante o desenvolvimento é o mais apropriado ao permitir a identificação e correção imediata dos erros.

5.7 Considerações Finais

Neste capítulo, foi apresentada uma avaliação experimental da abordagem proposta para representação visual e detecção de erros em regras SWRL. Esta avaliação foi constituída de um estudo de caso com o conjunto de regras *Autism Phenologue Rules*, um conjunto grande e complexo com 156 regras.

A abordagem inclui uma API (SWRL Editor API) que permite obter informações gerais e específicas sobre os átomos, predicados e argumentos das regras, além de detalhes sobre os tipos de átomos e argumentos. Isso nos possibilitou ter uma visão geral do conjunto de regras e de regras específicas. Para auxiliar na compreensão das regras, as representações visuais se mostraram adequadas possibilitando inclusive a geração de uma nova abstração para representar as regras deste conjunto específico.

A representação visual do autismo oculta detalhes técnicos da linguagem SWRL e representa adequadamente as regras sem perda de significado. Para tanto, foram identificados grupos de regras com estruturas comuns, os padrões dessas estruturas e, consequentemente, as

partes específicas que as diferenciam entre si. Esta foi uma das contribuições importantes deste trabalho.

A análise de similaridade serviu para comprovar as suposições sobre o quanto similares são as regras do autismo. Embora o método não ofereça suporte semântico, a técnica proposta e suas variações atendem aos requisitos iniciais, permitindo, por exemplo, o agrupamento de regras. O algoritmo de agrupamento K-means foi aplicado para comparar o método de classificação por meio da similaridade entre as regras ao invés do uso da estrutura das regras. A análise de similaridade dá suporte à detecção de erros, um requisito muito importante no gerenciamento de regras SWRL.

Por fim, a visualização e compreensão das regras foram melhoradas pelo uso da ferramenta SWRL Editor, que permitiu encontrar facilmente as regras (usando o mecanismo de busca), explorar a similaridade entre elas, explorar os agrupamentos obtidos e detectar erros para fins de validação ou simples conferência.

6 Conclusões

A linguagem SWRL permite aumentar a expressividade das ontologias definidas em OWL, por meio de afirmações condicionais (regras), proporcionando a inferência de novos conhecimentos sobre indivíduos da ontologia. Embora a sintaxe da linguagem SWRL seja simples para profissionais da computação, o mesmo não é válido para especialistas de outros domínios, no caso deste trabalho, pesquisadores biomédicos envolvidos no projeto *Phenologue*. Isso é verdade principalmente quando os conjuntos de regras usados se tornam grandes e complexos, aí mesmo desenvolvedores da área de computação tem dificuldades em gerenciá-las adequadamente.

As soluções desenvolvidas e apresentadas no decorrer deste trabalho, tiveram como principal objetivo contribuir para a melhoria da visualização e composição de regras SWRL, ao gerar representações visuais para estas regras, permitir a identificação de regras similares, o agrupamento a partir da similaridade e refinamentos na detecção de erros. Tais soluções foram integradas a ferramenta *Web SWRL Editor*, desenvolvida como um *plug-in* do Web Protégé, disponível em <http://biomac.icmc.usp.br:8080/SWRLEditor/>.

A seguir, são apresentadas as contribuições deste trabalho (Seções 6.1 e 6.2), as suas limitações e direcionamentos para trabalhos futuros (Seção 6.3).

6.1 Contribuições

O objetivo central deste trabalho foi desenvolver soluções que aprimorassem o processo de visualização e composição de regras SWRL. Especialmente para uso dos pesquisadores do projeto *Phenologue*. Para atingir esse objetivo, foram explorados 4 tópicos do trabalho com regras SWRL, para cada um deles apresentamos as contribuições desse trabalho.

6.1.1 Visualização de regras SWRL

Os objetivos deste trabalho aqui eram o desenvolvimento de representações visuais que abstraíam a sintaxe da linguagem SWRL; de um método de agrupamento que considere a

similaridade das regras; e a geração de informações automáticas, que forneçam uma visão geral e específica, de regras e de conjunto de regras.

As contribuições neste tópico são:

- **Lista de interfaces e recursos utilizados nas ferramentas de regras** – a partir do levantamento das ferramentas de regras, foi possível identificar e analisar as distintas formas de interação do usuário com as regras. Assim, as ferramentas, interfaces e recursos descritos no Capítulo 3 ilustram o cenário atual de desenvolvimento das ferramentas de regras, especialmente as que suportam a linguagem SWRL. As interfaces identificadas compreendem: tabelas e árvores de decisão; diagrama gráfico; editor de texto; formulários e *templates*; interface interativa; e linguagem natural. Além disso, recursos como segmentação de regras, representações visuais, agrupamento, componentes de sugestão e ícones complementam e enriquecem as interfaces e, por consequência, as mais diversas formas de interação;
- **Representações visuais das regras SWRL** – foram elaboradas três representações para abstrair as regras SWRL: *SWRL Highlight*, visualização hierárquica; e, visualização do autismo. Enquanto as duas primeiras são genéricas e podem ser empregadas para representar qualquer regra SWRL, a visualização do autismo é específica para as regras da ontologia *Autism Phenologue Rules* e oculta detalhes técnicos da linguagem SWRL sem perda de significado. Tais representações são destinadas primeiramente para simplificar a visualização, contudo, podem ser também empregadas para auxiliar na composição das regras. Além disso, ao mesmo tempo em que simplifica a apresentação da regra, apresenta informações adicionais sobre os tipos dos átomos. As representações visuais desenvolvidas são apresentadas nas Seções 4.4 e 5.3;
- **Agrupamento de regras por meio da similaridade entre regras SWRL** – com o objetivo de mensurar o quanto duas regras são similares (ou diferentes) a técnica desenvolvida utiliza a ocorrência dos átomos/predicados das regras para o cálculo do fator de similaridade. Com isso, quanto menor o valor obtido mais as regras são similares, de modo que, o valor 0 (zero) indica regras idênticas. Foram apresentadas variações da técnica com o intuito de atender a um maior número de cenários de uso. Uma das aplicações dessa estratégia consiste no agrupamento das regras a partir da similaridade entre elas (o uso de similaridade no agrupamento se mostrou mais adequado do que a utilização da estrutura da regra). A similaridade é abordada nas Seções 4.5 e 5.4, e o agrupamento nas Seções 4.6 e 5.5;

6.1.2 Composição das regras SWRL

Neste caso, os nossos objetivos eram a elaboração de uma lista com os possíveis erros (classificados em escopo e tipo) que os desenvolvedores estão sujeitos e implementação de um mecanismo que faça a detecção desses erros.

As contribuições neste tópico são:

- **Elaboração de uma lista categorizada de erros em regras SWRL** – foi elaborada uma lista de erros dividida em quatro categorias: (1) avisos; (2) erros léxicos; (3) erros sintáticos; e (4) erros semânticos. Para a elaboração da lista, foram considerados os estudos realizados e a observação das ferramentas de regras, sem deixar de atender a especificação da linguagem SWRL. A lista dos erros devidamente classificados é apresentada na Tabela 9, Seção 4.7;
- **Organização e detecção de erros em conjuntos de regras SWRL** – a partir da lista com os possíveis erros que os desenvolvedores de regras podem cometer, o mecanismo de detecção de erros foi desenvolvido. Apenas os erros classificados no tipo semântico não são suportados. Detalhes a respeito desta contribuição foram descritos nas Seções 4.7 e 5.6;

6.1.3 Colaboração entre os usuários

O objetivo aqui era permitir o desenvolvimento colaborativo e paralelo das regras SWRL. Para isso, foi desenvolvida uma ferramenta Web, como um *plug-in* do Web Protégé.

A contribuição neste tópico é:

- **Ferramenta SWRL Editor** – o *SWRL Editor* integra um conjunto de soluções para aprimorar a visualização e composição de regras SWRL. Embora não esteja ainda totalmente completa, é possível utilizar os principais recursos propostos neste trabalho. O grande diferencial do *SWRL Editor* em relação às ferramentas SWRL Tab, Axiomé e ACE View são o fato dele estar disponível na Web, seu mecanismo de busca de regras, a detecção de erros e as representações visuais que ele usa.. A ferramenta *SWRL Editor*, juntamente com as funcionalidades desenvolvidas, é apresentada na Seção 4.8;

6.1.4 Estudo de caso

O objetivo foi conduzir um estudo aplicando as técnicas desenvolvidas às regras da ontologia *Autism Phenologue Rules* para identificar os padrões, problemas e características desse conjunto de regras. Com isso, podemos validar algumas das técnicas empregadas e contribuir com os pesquisadores biomédicos envolvidos em seu desenvolvimento.

A contribuição neste tópico é:

- **Melhoramento do conjunto de regras SWRL da ontologia *Autism Phenologue Rules*** – a API e ferramenta desenvolvida permitiram analisar as regras dessa ontologia. Conduzida no Capítulo 5, a análise parte da observação de informações como a quantidade, média e distribuição dos átomos, predicados e argumentos no antecedente, consequente ou ambos. Em seguida, as representações visuais foram observadas permitindo encontrar os padrões que este conjunto de regras possui. As técnicas de análise de similaridade, agrupamento e detecção de erros foram aplicadas a essas regras e revelaram alguns problemas sobre as mesmas. Destaca-se ainda que, após a condução deste estudo de caso, houve um melhoramento desse conjunto de regras, pois: (1) problemas foram solucionados; (2) os desenvolvedores se motivaram a simplificar as regras; e (3) uma nova versão do conjunto de regras foi desenvolvida.

6.2 Publicações e Prêmios relacionados ao trabalho

Como resultados deste trabalho, foram aceitos para publicação os seguintes artigos:

Artigo publicado no ICEIS 2011 - *International Conference on Enterprise Information Systems* : “*An Analysis of Rules-Based Systems to Improve SWRL Tools*” (Rivolli, Orlando e Moreira, 2011);

Artigo publicado no WebMedia 2011 – Simpósio Brasileiro de Sistemas Multimídia e Web : “Regras SWRL: Análise de similaridade e detecção de erros” (Rivolli et al., 2011a);

Artigo publicado no WebMedia 2011 – Simpósio Brasileiro de Sistemas Multimídia e Web – Workshop de Ferramentas e Aplicações (WFA) : “Ferramenta web para visualização e composição de regras SWRL” (Rivolli et al., 2011b);

Artigo publicado no WebMedia 2011 – Simpósio Brasileiro de Sistemas Multimídia e Web – Workshop de Tese e Dissertação (WTD) : “Aprimorando a visualização e composição de regras SWRL na Web” (Rivolli e Moreira, 2011);

Prêmio de melhor ferramenta do X Workshop de Ferramentas e Aplicações do XVII Simpósio Brasileiro de Sistemas Multimídia e Web : “Ferramenta web para visualização e composição de regras SWRL” (Rivolli et al., 2011b);

6.3 Trabalhos Futuros

Como trabalhos futuros, podem ser consideradas as limitações e restrições encontradas no desenvolvimento deste trabalho. Podem ser listados como trabalhos futuros:

- Uma **avaliação de usabilidade** nas interfaces da ferramenta SWRL *Editor* e assim validar quantitativamente e qualitativamente os recursos desenvolvidos e o impacto que esses causam na interação com o usuário. De modo especial, realizar um estudo comparativo entre o uso das ferramentas de regras SWRL e identificar as melhores estratégias para visualização e composição das anotações.
- **Suporte à linguagem natural:** o suporte à linguagem natural se mostra um importante recurso para a melhoria do uso da ferramenta, uma vez que esse tipo de interface é a mais natural para usuários não técnicos em computação. Para tanto, pode-se utilizar a linguagem ACE.
- **Implementação de todos os recursos previstos na ferramenta SWRL Editor:** para completar a visualização das regras e do conjunto de regras, a árvore de decisão pode ser aplicada e tem grande potencial para cumprir este objetivo. Na composição foi previsto a sugestão de termos, contudo, alguns desenvolvedores consultados revelaram que o recurso de auto completar vem a ser mais atrativo que a sugestão. Por fim, a customização da ferramenta se mostra um importante recurso para torná-la familiar aos usuários.
- **Deteção de erros semânticos:** É desejável ter suporte semântico na análise de similaridade e na detecção de erros. Para tanto, no primeiro caso a matriz de características pode ser alterada para considerar informações semântica adicionais sobre os predicados disponíveis na ontologia: como hierarquia das classes e características das propriedades (transitividade, propriedades inversas, domínios,

faixa de valores, etc). Na detecção de erros, além de considerar as informações adicionais disponíveis na ontologia, é necessário recorrer a uma máquina de inferência para identificar possíveis equivalências e inconsistências lógicas.

- **Agrupamento hierárquico de regras:** Com relação à técnica de agrupamento, o emprego de outro algoritmo que se baseie na similaridade, porém que utilize uma estrutura hierárquica e seja determinístico, é sugerido. Com isso, os grupos podem ser melhores distribuídos e a reaplicação do método não gera novos grupos.
- **Elaboração de uma metodologia genérica formal que utiliza as técnicas propostas neste trabalho:** a análise do conjunto de regras se mostrou completa o bastante para ser generalizada em um modelo de análise e estudo de regras SWRL. O resultado das técnicas de similaridade somado ao conjunto de informações disponibilizado pela ferramenta possibilita também classificar os conjuntos, por exemplo, em homogêneo/heterogêneo e simétrico/assimétrico.

Referências

ALMEIDA, M. B.; BAX, M. P. Uma visão geral sobre ontologias: pesquisa sobre definições, tipos, aplicações, métodos de avaliação e de construção. **Ci. Inf.**, Brasília, v. 32, n. 3, Dec. 2003. doi: 10.1590/S0100-19652003000300001.

BADA, M.; TURI, D.; McENTIRE, R.; STEVENS, R. 2004. Using reasoning to guide annotation with gene ontology terms in GOAT. **SIGMOD Rec.**, New York, NY, USA, v. 33, n. 2, p. 27-32, Jun. 2004. doi: 10.1145/1024694.1024699.

BARDER, F.; NUTT, W. Basic Description Logics. In: BARDER, F. et al. **The Description Logic Handbook**. Cambridge University Press. 2003. capítulo 2, p. 43-90.

BERNERS-LEE, T.; HENDLER, J.; LASSILA, O. The Semantic Web. **Scientific American**, p. 34-43, 2001. Disponível em: <http://www.scientificamerican.com/article.cfm?id=the-semantic-web>, acesso em Out. 2009.

BRAYE, L.; RAMEL, S.; GRÉGOIRE, B.; LEIDNER, S.; SCHMITT, M. **State of the Art Business Rules Languages**. Public Research Centre Henri Tudor, 2006. Disponível em: [http://efficient.citi.tudor.lu/cms/efficient/content.nsf/0/4A938852840437F2C12573950056F7A9/\\$file/BusinessRulesLanguages_D3.1.pdf](http://efficient.citi.tudor.lu/cms/efficient/content.nsf/0/4A938852840437F2C12573950056F7A9/$file/BusinessRulesLanguages_D3.1.pdf), acesso em Jun. 2010.

BRICKLEY, D.; GUHA, R. V.; McBRIDE, B. (Editors). **RDF Vocabulary Description Language 1.0: RDF Schema**. W3C. Fev. 2004. Disponível em: <http://www.w3.org/TR/rdf-schema/>, acesso em Jan. 2010.

CHANDRASEKARAN, B.; JOSEPHSON, J. R.; BENJAMINS, V. R. What Are Ontologies, and Why Do We Need Them?. **IEEE Intelligent Systems**, Piscataway, NJ, USA, v. 14, n. 1, p. 20-26. Jan. 1999. doi: 10.1109/5254.747902.

CORAZZON, R. Theory and History of Ontology. **A Resource Guide for Philosophers**. 2010. Disponível em: <http://www.formalontology.it/>, acesso em Jan. 2010.

DINIZ, V.; CECCONI, C. **Padrões Web: Passado, presente e futuro**, V Conferência Latino Americana de Software Livre. 2008. Disponível em: http://www.w3c.br/palestras/internet-web-jun-jul-2008/internetWeb_Out08.html, acesso em Dez. 2009.

EILBECK, K.; LEWIS, S. E. Sequence Ontology Annotation Guide. **Comp Funct Genomics**, v. 5, n. 8, p. 642-647. Dez. 2004. doi: 10.1002/cfg.446.

ERDMANN, M.; MAEDCHE, A.; SCHNURR, H. P.; STAAB, S. From manual to semi-automatic semantic annotation: About ontology-based text annotation tools. In: **Proceedings of the COLING 2000 Workshop on Semantic Annotation and Intelligent Content**, Luxemburgo. Ago. 2000. doi: 10.1.1.2.4325.

FEIGENBAUM, L.; HERMAN, I.; HONGSERMEIER, R.; NEUMANN, E.; STEPHENS, S. **The Semantic Web in action**, *Scientific American*, p. 64-71, Dez. 2007. Disponível em: <http://www.ncbi.nlm.nih.gov/pubmed/18237102>, acesso em Jan. 2010.

FENSEL, D.; VAN HARMELEN, F.; HORROCKS, I.; McGUINNESS, D. L.; PATEL-SCHNEIDER, P. F. OIL: An Ontology Infrastructure for the Semantic Web. **IEEE Intelligent Systems**, v. 16, n. 2, p. 38-45. Abr. 2001. doi: 10.1109/5254.920598.

FRIEDMAN-HILL, E. **Jess the Rule Engine for the Java™ Platform**. Sandia National laboratories, versão 7.1, Jul. 2008. disponível em <http://herzberg.ca.sandia.gov/docs/Jess71.pdf>, acesso em Jun. 2011.

FRITH, U.; HILL, E. L. **Autism: Mind and Brain**. Oxford University Press. 2003.

GOLBREICH, C.; HORRIDGE, M.; HORROCKS, I.; MOTIK, B.; SHEARER, R. OBO and OWL: leveraging semantic web technologies for the life sciences. In: **Proceedings of the 6th international The semantic web and 2nd Asian conference on Asian semantic web conference**. Busan, Korea, p. 169-182. 2007.

GRAU, B. C.; HORROCKS, I.; MOTIK, B.; PARSIA, B.; PATEL-SCJMEODER, P.; SATTLER, U. OWL 2: The next step for OWL. **Journal of Web Semantics: Science, Services and Agents on the World Wide Web**, Amsterdam, Netherlands, v. 6, n. 4, p. 309-322. Nov. 2008. doi: 10.1016/j.websem.2008.05.001.

GRUBER, T. R. A translation approach to portable ontology specifications. **Knowl. Acquis.** London, UK, v. 5, n. 2, p. 199-220. Jun. 1993. doi: 10.1006/knac.1993.1008.

GUARINO, N.; GIARETTA, P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. **Journal Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing**. v. 1, n. 9, p. 25-32. 1995.

HAAV, H. M.; LUBI, T. L. A survey of concept-based information retrieval tools on the web. In: **Proceedings of East-European Conference ADBIS**. v. 2, p. 29-41. 2001.

HASSANPOUR, S.; O'CONNOR, M. J.; DAS, A. K. Exploration of SWRL Rule Bases through Visualization, Paraphrasing, and Categorization of Rules. In: **Proceedings of the 2009 International Symposium on Rule Interchange and Applications (RuleML 2009)**, Las Vegas, Nevada, p. 246-261. 2009. doi: 10.1007/978-3-642-04985-9_23.

HASSANPOUR, S.; O'CONNOR, M. J.; DAS, A. K. Visualizing Logical Dependencies in SWRL Rule Bases. **The International RuleML Symposium on Rule Interchange and Applications**, Washington, DC, p. 259-272. 2010.

HORRIDGE, M.; BECHHOFFER, S. The OWL API: A Java API for Working with OWL 2 Ontologies, In: **Proceedings of the OWL Experienced and Directions Workshop (OWLED 2009)**, Chantilly, Virginia. 2009. doi: 10.1.1.163.7035.

HORRIDGE, M.; KNUBLAUCH, H.; RECTOR, A.; STEVENS, R.; WROE, C. **A practical guide to building OWL ontologies using the protégé-OWL plugin and CO-ODE tools edition 1.0**. University Of Manchester. 2004. Disponível em: <http://www.co-ode.org/resources/>, acesso em Out. 2009.

HORROCKS, I.; PATEL-SCHNEIDER, P. F.; BOLEY, H.; TABET, S.; GROSOFF, B.; DEAN, M. **SWRL: A Semantic Web Rule Language Combining OWL and RuleML**. W3C. Maio 2004. Disponível em: <http://www.w3.org/Submission/SWRL/>, acesso em Jan. 2010.

JAIN, A. K.; MURTY, M. N.; FLYNN, P. J. Data clustering: A review. **ACM Computer Survey**, New York, NY, USA, v. 31, n. 3, p. 264-323. Set. 1999. doi: 10.1145/331499.331.

JONQUET, C.; MUSEN, M. A.; SHAH, N. A System for Ontology-Based Annotation of Biomedical Data. In: **Proceedings of the 5th international workshop on Data Integration in the Life Sciences (DILS '08)**, Evry, France, p. 144-152. 2008. doi: 10.1007/978-3-540-69828-9_14.

KALJURAND, K. ACE View – an ontology and rule editor based on Attempto Controlled English. In: **Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions (OWLED 2008)**, v. 432, p. 1-12, Out. 2008. doi: 10.5167/uzh-8822.

KIRYAKOV, A.; POPOV, B.; TERZIEV, I.; MANOV, D.; OGNYANOFF, D. Semantic annotation, indexing, and retrieval. **Web Semantics: Science, Services and Agents on the World Wide Web** 2, v. 2, n. 1, p. 49-79. Dez. 2004. doi: 10.1016/j.websem.2004.07.005.

KLYNE, G.; CARROL, J. J.; McBRIDE, B. (Editors). **Resource Description Framework (RDF): Concepts and Abstract Syntax**. W3C. Fev. 2004. Disponível em: <http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/>, acesso em Set. 2009.

KNUBLAUCH, H.; FERGERSON, R. W.; NOY, N. F.; MUSEN, M. A. The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. **ISWC 2004: Third International Semantic Web Conference**. Hiroshima, Japan, p. 229-243. 2004. doi: 10.1.1.89.9268.

LASSILA, O.; SWICK, R. (Editors). **Resource Description Framework (RDF) model and syntax specification**. W3C. Fev. 2004. Disponível em: <http://www.w3.org/TR/REC-rdf-syntax/>, acesso em Out. 2009.

McGUINNESS, D. L.; HARMELEN, F. (Editors). **OWL Web Ontology Language Overview**. W3C. Fev. 2004. Disponível em: <http://www.w3.org/TR/2004/REC-owl-features-20040210/>, acesso em Jun. 2009.

NDAR. **Frequently Asked Questions**. 2010. Disponível em: <http://ndar.nih.gov/ndarpublicweb/faq.go>, acesso em Fev. 2011.

NOY, N. F.; McGUINNESS, D. L. **Ontology Development 101: A Guide to Creating Your First Ontology**. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05. 2001.

O'CONNOR, M. J.; MUSEN, M. A.; DAS, A. K. Using the Semantic Web Rule Language in the Development of Ontology-Driven Applications. In: **Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches**, Hershey, Pennsylvania, p. 525-539. Capítulo XXII, 2009.

O'CONNOR, M. J.; DAS, A. K. SQWRL: a Query Language for OWL. **OWL: Experiences and Directions (OWLED 2009)**, Chantilly, VA, v. 271, p. 3-22. 2009. doi: 10.1016/j.entcs.2011.02.008.

OUELLET, R.; OGBUJI, U. **Introduction to DAML: Part I**. Jan. 2002. Disponível em: <http://www.xml.com/pub/a/2002/01/30/daml1.html>, acesso em Jan. 2010.

PEACE, J.; BRENNAN, P. F. Instance testing of the family history ontology. In: **Proceedings of the American Medical Informatics Association (AMIA) Annual Symposium**. Nov. 2008.

RIVOLLI, A.; MOREIRA, D. A. Aprimorando a visualização e composição de regras SWRL na Web. **XVII WebMedia: Simpósio Brasileiro de Sistemas Multimídia e Web: Workshop de teses e dissertações**, Florianópolis, SC. Out. 2011.

RIVOLLI, A.; ORLANDO, J. P.; MOREIRA, D. A. An Analysis of Rules-Based Systems to Improve SWRL Tools. **ICEIS 2011: International Conference on Enterprise Information Systems**, Beijing, China, p. 191-194. Jun. 2011.

RIVOLLI, A.; ORLANDO, J. P.; SERIQUE, K.; MOREIRA, D. A. Ferramenta web para visualização e composição de regras SWRL. **XVII WebMedia: Simpósio Brasileiro de Sistemas Multimídia e Web: Workshop de ferramentas e aplicações**, Florianópolis, SC. Out. 2011a.

RIVOLLI, A.; ORLANDO, J. P.; YAMAMOTO, C. H.; MOREIRA, D. A. Regras SWRL: Análise de similaridade e detecção de erros. **XVII WebMedia: Simpósio Brasileiro de Sistemas Multimídia e Web**, Florianópolis, SC. Out. 2011b.

RUBIN, D. L.; NOY, N. F.; MUSEN, M. A. Protégé: A Tool for Managing and Using Terminology in Radiology Applications. In: *Journal of Digital Imaging*, v. 20, n. 1, p. 34-46. Nov. 2007. doi: 10.1007/s10278-007-9065-0.

SALZBERG, S. Distance Metrics for Instance-Based Learning. In: **Proceedings of ISMIS'916th International Symposium**, v. 542, p. 399-408. 1991. doi: 10.1007/3-540-54563-8_103.

SHADBOLT, N.; HALL, W.; BERNERS-LEE, T. The Semantic Web Revisited. **IEEE Intelligent Systems**, v. 21, n. 3, p. 96-101. Jun. 2006. doi: 10.1109/MIS.2006.62.

SHAH, N. H.; MUSEN, M. A. **Which Annotation did you mean?**. Stanford Center for Biomedical Informatics Research. Set. 2007. Disponível em: http://bmir.stanford.edu/publications/view.php/which_annotation_did_you_mean, acesso em Dez. 2009.

SMITH, B.; WELTY, C. Ontology: Towards a New Synthesis. In: **FOIS'01: Proceedings of the international conference on Formal Ontology in Information Systems**, Ogunquit, Maine, USA, p. 3-9. 2001. doi: 10.1145/505168.505201.

SMITH, M. K.; WELTY, C.; MCGUINNESS, D. L. (Editors). **OWL Web Ontology Language Guide**. W3C. Fev. 2004. Disponível em: <http://www.w3.org/TR/owl-guide/>, acesso em Nov. 2008.

STAAB, S.; MAEDCHE, A.; HANDSCHUH, S. An annotation framework for the semantic web. In: **Proceedings of the First Workshop on Multimedia Annotation**, Tokyo, Japan, p. 30-31. 2001. doi: 10.1.1.25.910.

SU, X.; ILEBREKKE, L. A Comparative Study of Ontology Languages and Tools. In: **CAiSE'02: Proceeding of the 14th Conference on Advanced Information Systems Engineering**, Toronto, Canada, v. 2348, p. 761-765. 2002. doi: 10.1007/3-540-47961-9_62.

TU, S.; TENNAKON, L.; O'CONNOR, M. J.; SHANKAR, R.; DAS, A. K. Using an integrated ontology and information model for querying and reasoning about phenotypes: the case of autism. In: **Proceedings of the American Medical Informatics Association**, Washington, DC, v. 6, p. 727-731. Nov. 2008.

TUDORACHE, T.; VENDETTI, J.; NOY, N. F. Web-Protégé: A Lightweight OWL Ontology Editor for the Web. In: **OWLED 2008: OWL: Experiences and Directions**, Karlsruhe, Germany. Out. 2008. doi: 10.1.1.142.8568.

USCHOLD, M. 1998. Knowledge level modelling: concepts and terminology. **The Knowledge Engineering Review**, v. 13, n. 1, p. 5-29. Mar. 1998. doi: 10.1017/S0269888998001040.

USCHOLD, M.; GRÜNINGER, M. Ontologies: principles, methods and applications. **The Knowledge Engineering Review**, v. 11, n. 2, p. 93-155. 1996. doi: 10.1017/S0269888900007797.

W3C OWL Working Group. **OWL 2 Web Ontology Language Document Overview**. W3C. Out. 2009. Disponível em: <http://www.w3.org/TR/owl2-overview/>, acesso em Dez. 2009.

ZACHARIAS, V. Development and verification of rule based systems – a survey of developers. In: **Rule Representation, Interchange and Reasoning on the Web: International Symposium**, Orlando, Florida, USA, v. 5321, p. 6-16, 2008. doi:10.1007/978-3-540-88808-6_4.

Apêndice A – Representações visuais alternativas

SWRL Highlight 1

```

ADI-2003(?a) ^
adi-r2003:ADI_2003_cbacs_atotal(?a, ?score) ^
swrlb:greaterThan(?score, 10) ^
adi-r2003:SubjectKey(?a, ?subjectID) ^
adi-r2003:ADI_2003_interview_date(?a, ?date) ^
swrlx:createOWLThing(?phenorecord, ?subjectID)
->
'Phenotype record'(?phenorecord) ^
temporal:hasValidTime(?phenorecord, ?date) ^
'is derived from'(?phenorecord, "ConcludeADIReciprocalSocialInteraction") ^
autism-core:subjectId(?phenorecord, ?subjectID) ^
autism-core:subject_has_quality_or_disposition(?phenorecord, 'ADI
qualitative abnormalities in reciprocal social interactions status')

```

SWRL Highlight 2

```

ADI-2003(?a) ^
adi-r2003:ADI_2003_cbacs_atotal(?a, ?score) ^
swrlb:greaterThan(?score, 10) ^
adi-r2003:SubjectKey(?a, ?subjectID) ^
adi-r2003:ADI_2003_interview_date(?a, ?date) ^
swrlx:createOWLThing(?phenorecord, ?subjectID)
->
'Phenotype record'(?phenorecord) ^
temporal:hasValidTime(?phenorecord, ?date) ^
'is derived from'(?phenorecord, "ConcludeADIReciprocalSocialInteraction") ^
autism-core:subjectId(?phenorecord, ?subjectID) ^
autism-core:subject_has_quality_or_disposition(?phenorecord, 'ADI
qualitative abnormalities in reciprocal social interactions status')

```

SWRL Highlight 3

```

ADI-2003(?a) ^
adi-r2003:ADI_2003_cbacs_atotal(?a, ?score) ^
swrlb:greaterThan(?score, 10) ^
adi-r2003:SubjectKey(?a, ?subjectID) ^
adi-r2003:ADI_2003_interview_date(?a, ?date) ^
swrlx:createOWLThing(?phenorecord, ?subjectID)
->
'Phenotype record'(?phenorecord) ^
temporal:hasValidTime(?phenorecord, ?date) ^
'is derived from'(?phenorecord, "ConcludeADIReciprocalSocialInteraction") ^
autism-core:subjectId(?phenorecord, ?subjectID) ^
autism-core:subject_has_quality_or_disposition(?phenorecord, 'ADI
qualitative abnormalities in reciprocal social interactions status')

```

SWRL Highlight 4

```

ADI-2003(?a) ^
adi-r2003:ADI_2003_cbacs_atotal(?a, ?score) ^
swrlb:greaterThan(?score, 10) ^
adi-r2003:SubjectKey(?a, ?subjectID) ^
adi-r2003:ADI_2003_interview_date(?a, ?date) ^
swrlx:createOWLThing(?phenorecord, ?subjectID)
->
'Phenotype record'(?phenorecord) ^
temporal:hasValidTime(?phenorecord, ?date) ^
'is derived from'(?phenorecord, "ConcludeADIReciprocalSocialInteraction") ^
autism-core:subjectId(?phenorecord, ?subjectID) ^
autism-core:subject_has_quality_or_disposition(?phenorecord, 'ADI
qualitative abnormalities in reciprocal social interactions status')

```

SWRL Visualization 1

SWRL Visualization 2

SWRL Visualization 2

Apêndice B – Protótipos de interface para a ferramenta SWRL Editor

Estrutura principal da ferramenta

Opções do painel *Rule Properties*

Painéis *Rule Properties* e *Rule Painel* no modo de visualização

Painel *Rule Painel* com a aba *List* ativa e as possibilidades de visualização

Painel *Rule* Painel com a aba *Groups* ativa

Painel *Rule Painel* com a aba *Decision Tree* ativa

Painéis *Rule Properties* e *Rule Painel* no modo de composição

Painel *Rule Painel* no modo de composição com a opção *Rule Editor* ativa

Apêndice C – Estrutura comum identificada nas regras do autismo

ESTRUTURA COMUM C

IF

●
?a
Vineland Survey Interview -
2005

■ vabs_survey:SubjectKey ?subjectID
■ vabs_survey:Vinelandsurvey_2005_interview_date ?date

■ swrlx:createOWLThing(?phenorecord, ?subjectID)
■ swrlx:createOWLThing(?phenorecord, ?qv)

THEN

●
?phenorecord
Phenotype
record

■ autism-core:subject_has_quality_or_disposition
■ temporal:hasValidTime ?date
■ has measurement ?qv
■ is derived fromTime ?date
■ autism-core:subjectId ?subjectID

●
?qv
Quantitative
value

■ has value ?score

ESTRUTURA COMUM D

IF

●
?a
ADOS-1 2001

■ ados1:SubjectKey ?subjectID
■ ados1:Ados1_2001_interview_date ?date

■ swrlx:createOWLThing(?phenorecord, ?subjectID)
■ swrlx:createOWLThing(?phenorecord, ?qv)

THEN

●
?phenorecord
Phenotype
record

■ autism-core:subject_has_quality_or_disposition
■ temporal:hasValidTime ?date
■ has measurement ?qv
■ is derived fromTime ?date
■ autism-core:subjectId ?subjectID

●
?qv
Quantitative
value

■ has value ?score

ESTRUTURA COMUM E

IF

● ?a ADOS-1 2001	■ ados4:SubjectKey ?subjectID ■ ados4:Ados4_2001_interview_date ?date
------------------------	--

■ swrlx:createOWLThing(?phenorecord, ?subjectID)

■ swrlx:createOWLThing(?phenorecord, ?qv)

THEN

● ?phenorecord Phenotype record	■ autism-core:subject_has_quality_or_disposition ■ temporal:hasValidTime ?date ■ has measurement ?qv ■ is derived fromTime ?date ■ autism-core:subjectId ?subjectID
--	---

● ?qv Quantitative value	■ has value ?score
-----------------------------------	--------------------